ACCELERATING MATLAB…DEVICES THAT BLOG…ECLIPSE CDT PROJECT

Embedded COMPUTING DESIGN

www.embedded-computing.com

NOVEMBER 2005 VOLUME 3 NUMBER 6

Windows CE and Windows XP in real time

OSDL UPDATE

OSDL's new embedded Linux initiative

APPLICATION FEATURE

Biometrics security through middleware

PRODUCT GUIDE FEATURE

Embedded test environments

PICTURED
Lantronix Xport AR device server

Embedded COMPUTING DESIGN

www.embedded-computing.com VOLUME 3 • NUMBER 6 NOVEMBER 2005

COLUMNS

COVER
Authors reflect on Windows for real time.

PRODUCT
The Lantronix XPort AR includes a complete embedded computer, an Ethernet 10/100 interface along with an onboard network operating system, and web server in a miniaturized RJ-45 package about the size of two sugar cubes, and enables manufacturers to quickly and easily build Ethernet or IP networking capabilities into their products.

Published by: OpenSystems Publishing™

FEATURES

APPLICATION

SPECIAL: RTOS

PRODUCT GUIDE: Embedded Test

E-LETTER

www.embedded-computing.com/eletter
Learn how the world's smallest silicon TV tuner can enhance your embedded display systems
By Xceive Inc.

EVENTS

ESC Silicon Valley
April 3-7, 2006 • San Jose, CA • McEnery Convention Center
www.esconline.com

WEB RESOURCES

Subscribe to the magazine or E-letter:
www.opensystems-publishing.com/subscriptions

Industry news:
Read: www.embedded-computing.com/news
Submit: www.opensystems-publishing.com/news/submit

Submit new products:
www.opensystems-publishing.com/vendors/submissions/np

Real-time Windows and development tools


Jerry Gipper

Welcome to the November issue of *Embedded Computing Design*. I am excited to be the editorial director of this publication. Embedded computing has been a passion of mine for my entire career. This is a great opportunity for me to learn even more about this industry and share that knowledge with you. I look forward to informing you about the latest news, trends, and issues involving embedded computing designs.

In this issue we will be covering:

Real-Time Operating Systems and Windows

- *Extending Windows XP into real time*, by Paul Fischer of TenAsys. Paul talks about the various ways to add real-time support to a Windows XP environment. Dual operating systems, Windows XP, and a Real-Time Operating System can combine to provide a powerful operating environment for embedded applications. Paul also looks at the impact of new multi-core processors that may make it even more attractive to use a dual operating system configuration.
- *Windows CE 5.0 for real-time systems*, by Mike Hall of Microsoft. Windows CE was designed from the ground up to be a small footprint, componentized, hard real-time embedded operating. Mike analyzes the interrupt handling capabilities of Windows CE and some of the operating system tools used to verify Windows CE 5.0 real-time behaviors.

Embedded Linux

- *New OSDL initiative targets current and next-generation converged handsets*, by Bill Weinberg of OSDL. Bill updates us on some new initiatives underway by OSDL. Embedded Linux has a major impact on the embedded computing industry, and these initiatives could be important in future deployments of Linux.

Embedded biometrics

- *Biometrics security through middleware*, by Shoieb Yunus of EzValidation. Biometrics is becoming a more common solution to protecting our valuable information. This article discusses device security needs and presents middleware tools to add fingerprint-based security.

Diagnostic software

- *Gaining market advantage with an advanced embedded test environment*, by Joseph Skazinski of Kozio. Embedded test software is a key element to successful hardware products. Joseph points out the places where embedded test software is most useful, and explains why it is beneficial.
- *Creating fast, bit-accurate fixed-point simulation models in MATLAB for accurate verification and prototyping of signal processing algorithms*, by Anu Shivaprasad of Catalytic. Anu

continued on page 9

Embedded COMPUTING DESIGN
AN OPENSYSTEMS PUBLICATION

Embedded and Test & Analysis Group

- Embedded Computing Design
- Embedded Computing Design E-letter
- Embedded Computing Design Product Directory
- Industrial Embedded Systems Product Directory
- PCI Express Product Directory
- PXI Technology Review

Editorial Director	Jerry Gipper jgipper@opensystems-publishing.com
Contributing Editor	Don Dingee
Technical Editor	Chad Lumsden clumsden@opensystems-publishing.com
Associate Editor	Jennifer Hesse
European Representative	Hermann Strass hstrass@opensystems-publishing.com
Special Projects Editor	Bob Stasonis
Senior Designer	Joann Toth
Senior Web Developer	Konrad Witte
Graphic Specialist	David Diomede
Circulation/Office Manager	Phyllis Thompson subscriptions@opensystems-publishing.com

OpenSystems Publishing

Editorial/Production office:
16872 E. Avenue of the Fountains, Ste 203, Fountain Hills, AZ 85268
Tel: 480-967-5581 ■ Fax: 480-837-6466
Website: www.opensystems-publishing.com

Publishers	John Black, Michael Hopper, Wayne Kristoff
Vice President Editorial	Rosemary Kristoff

Communications Group

Editorial Director	Joe Pavlat
Assistant Managing Editor	Anne Fisher
Senior Editor (columns)	Terri Thorson
Technology Editor	Curt Schwaderer
Associate Editor	Jennifer Hesse
European Representative	Hermann Strass

Military & Aerospace Group

Group Editorial Director	Chris Ciufo
Managing Editor	Bonnie Crutcher
Assistant Editor	Sharon Schnakenburg
Senior Editor (columns)	Terri Thorson
European Representative	Hermann Strass
European Bureau Chief	Stefan Baginski

ISSN: Print 1542-6408, Online 1542-6459

Embedded Computing Design is published bi-monthly by OpenSystems Publishing LLC., 30233 Jefferson Ave., St. Clair Shores, MI 48082.

Subscriptions are free to persons interested in the design or promotion of embedded computing systems. For others inside the US and Canada, subscriptions are $34/year. For 1st class delivery outside the US and Canada, subscriptions are $60/year (advance payment in US funds required).

Canada: Publication agreement number 40048627
Return address: WDS, Station A, PO Box 54, Windsor, ON N9A 615

POSTMASTER: Send address changes to ***Embedded Computing Design***
16872 E. Avenue of the Fountains, Ste 203, Fountain Hills, AZ 85268

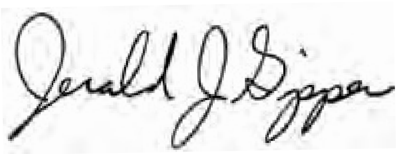
Save months of development time by using our ready-to-run Intel XScale® SBC's with your choice of operating system...
VULCAN single board computer
• 533MHz Intel® IXP425 network processor
• x2 100baseTx Ethernet ports
• Accelerated encryption hardware
• x4 USB 2.0 and x4 serial ports
• 64Mbytes DRAM / 32Mbytes Flash
• CompactFlash (CF+) and PC/104 bus
VIPER single board computer
• 400MHz Intel® PXA255 processor
• Ultra-low power 1.9W (typical)
• TFT/STN graphics controller
• 64Mbytes DRAM / 32Mbytes Flash
• Ethernet, USB and serial ports
• CompactFlash (CF+) and PC/104 bus
Embedded Linux Development Kit
• Based on 2.6 kernel
• Optional embedded Java technology
Windows® CE .NET/5.0 Development Kit
• Standard build includes comprehensive set of Windows CE components
• Includes Embedded Visual Tools C++ Tools
Combining the very best of low power Intel XScale® RISC technology and the industry standard PC/104 format.
VxWorks® Development Kit
• Support for VxWorks 5.5 real time OS
• WindML 3.0 multi-media library
QNX 6.3 Development Kit
• Support via third party BSP
CUSTOM DESIGN & BUILD
ARCOM
Use our Design & Build service to achieve cost effective custom design solutions.
Rugged industrial compact enclosure.
www.arcom.com
Development Kits
Embedded Boards
PCs & Enclosures
Industrial Networking
Design & Build
arcom
us-sales@arcom.com
888-941-2224
Think Embedded. Think Arcom.
RSC #8 @ www.embedded-computing.com/rsc

describes how to create processes for helping migrate from floating to fixed-point math in signal processing algorithms. Not using the right processes can create unacceptable behavior in the models used for simulation. This article gives an overview of recommended steps in creating good models.

Also, we are working on a new editorial calendar for the coming year. I look forward to bringing you new and interesting perspectives on the embedded computing industry. We are seeking interesting applications of embedded technologies, and in particular, the software environment needed for development and deployment of a successful embedded application.

I am also going to consider the impact of embedded computing. This is a pervasive technology that touches each one of us on a constant daily basis.

Your suggestions and comments are welcome. Please contact me at jgipper@opensystems-publishing.com.



Jerry Gipper
Editorial Director

jgipper@opensystems-publishing.com

EMBEDDED PERSPECTIVE

By Don Dingee

Have your device blog my device

We live in an age of instant, pervasive communication.

- Blogging
- Text messaging
- Podcasting
- Voicemail
- Flash mobs
- Camera phones
- MySpace

Everyone has something to say and wants to be heard – now. The problem is there's more being said than we possibly have time to hear.

Some folks have already figured out personalization and *pull* can be part of the solution. We express preferences used to filter the communication we'd like to receive, and presumably most of the stuff we get is of direct interest to us – or at least it's a much shorter list we can sort through quickly. In the endgame, most of us would probably like to send an agent out to gather the stuff of interest to us, and the agent would automatically get it, sort it, and present it to us on-demand on the device we want to use to review it.

The Internet of things

Designers have been scrambling to put every embedded device on the Internet, and not just devices that interface to humans, but every device that collects data about something. Several thought leaders have called this *the Internet of things*. Vinton Cerf of Google is predicting there will be between 5 and 20 billion Internet connected devices by 2010. That's 3 to 15 times the size of the current global telephone network, a staggering figure when fully considered.

The upside to this is more data is available about communications, defense, medical, industrial, and other embedded systems than ever before. The downside is, there's a lot more data – more than we possibly have time to sort through, and we're already overloaded.

But some of these devices have a lot of really important things to say, and they need to be heard. So let's go back to the idea of *pull*. Rather than having every embedded device spew over its connection, it'd be a lot better to have it automatically update us (or our agent, who then passes the information along to us with some filtering applied) when something important happens. We don't want to check that device, and the thousands of others we're interested in, every five minutes or five milliseconds depending on the real-time nature of the process.

Even better, some of these really smart devices could communicate information between each other while humans are busy listening to other conversations. The devices could exchange the needed data, make most of the decisions autonomously, and then notify us when something out of the ordinary needs our attention.

Applications are changing

Let's look at just one application, inventory control. These systems are changing rapidly as individual items are identified and reported back to centralized enterprise software via Ethernet. Here are two examples:

- Perfect Gate in Kansai, Japan, designs and develops access control systems to lower operating costs and provide better security for bicycle parking areas. When a customer brings a bike to the parking area, they and the bike each receive a unique Radio Frequency Identification (RFID) tag read on entry and exit. Based on info transmitted from the readers, the lot control system knows when a person entered and left the lot, what bike they brought in, and what bike they are allowed to take out. Billing is handled automatically (often through RFID-enabled *wallet phones*), and operational costs are reduced through minimizing lot staffing requirements.

- Bartech Systems International in Millersville, Maryland, offers an automated hotel mini-bar system. The e-Fridge is installed in more than 18,000 hotel rooms in the US and other locations worldwide. The mini-bar senses when an item is removed for more than 30 seconds, then sends a message over its Ethernet connection (wired or wireless are available) back to a management system in the hotel, which automatically compiles the billing and notifies the housekeeping staff of restocking requirements. When individual items become RFID enabled, this application could certainly benefit from positive identification of specific items. Imagine the level of service the hotel can provide if they knew my beverage preference, made sure my room was stocked, and then could report in aggregate on guest consumption trends back to their mini-bar item vendors. Innovations such as dynamic pricing and inventory are right around the corner.

Syndication is key

Along with RFID, another powerful tool available for embedded designers is Really Simple Syndication (RSS). RSS is based on a standardized XML (Extensible Markup Language) format, usually sending a headline and a short description along with a link to the full text. Articles are syndicated to readers subscribing to the feed.

I use RSS to glean news on several publicly traded companies and my favorite sports teams, pulling information from several syndicated sources. These news items are available in a consolidated, easily scanned headline format in my browser so I don't have to go chase down dozens of websites looking for new information. I can read the articles I want quickly and easily this way.

Let's say Mr. Cerf is on target (which he usually is), and in the near future up to 20 billion Ethernet-enabled devices are on the Web. Somewhere out on the network there are probably 10, 50, or maybe as many as a couple hundred devices that I really care about for my job. If I could *subscribe* to those devices via RSS, they could tell me exactly what I want to hear now. I can always change my subscription at any time as my interests and needs change. Or, I could design a device with RSS capabilities that could do the publishing or subscribing. Then my new device could blog with these other devices of interest, and my reach could be extended to thousands or even millions of devices.

Chris Humphrey, Sr. VP of Marketing at Lantronix in Irvine, CA put it this way – "enabling devices that blog" – in describing the new Lantronix XportAR. The Xport family has been around for a while, enabling designers to add Ethernet ports to devices quickly and easily. The Xport AR goes the next step with an embedded XML and RSS server along with other enabling software in their Evolution Network Operating System.

Lantronix views machine-to-machine connections as the next big breakthrough, coining the phrase *machine-to-mainstream*. This functionality is seen as providing OEMs a better way to harvest and analyze data from their embedded devices, and perhaps even enabling new revenue opportunities through access to subscribed data and better serviceability. Initial uses are seen in medical, industrial automation, building automation and security, and other embedded applications that touch the edges of the enterprise IT systems.

Embedded devices shouldn't just be a way to get on an Ethernet network. They should be part of the network, and smart enough to not just overload us with more data, but intelligently present the information we need in a concise, real-time format. With tools like RFID and RSS coming into play, soon we'll have devices blogging to other devices, and hopefully we'll all be more productive for it.

Speaking of which, you can use RSS right now to be part of the OpenSystems Publishing network. Industry News on our website at www.embedded-computing.com is an RSS feed you can subscribe to in the ongoing quest to keep up to date in real time as news is happening. And as always, I welcome your feedback and ideas, and you can e-mail me at ddingee@opensystems-publishing.com.

Eclipse Focus: CDT 3.0

Q & A with Doug Schaefer, QNX Software Systems, and Chris Recoskie, Texas Instruments

By Don Dingee and Chad Lumsden

ECD recently had the opportunity to pose a few questions to Doug Schaefer, the new Eclipse C Development Tools (CDT) Project Leader and Senior Software Developer at QNX Software Systems. Doug worked with a committer on the Eclipse CDT team, Chris Recoskie of Texas Instruments (TI), in answering some of our questions. In these responses, we gain some insights into the latest version of CDT 3.0 and how it is being used by software developers.

Q ECD: What advantages does CDT 3.0 offer for embedded developers in coding, optimizing, profiling, and debugging real-time code?

A Schaefer: We've implemented a number of usability improvements in CDT 3.0, with significant effort being placed on the CDT core's parser-based features, the managed build system, and the CDT debug views. The core features a faster parser and a complete abstract syntax tree, together called the DOM. This feature enables faster and more accurate refactoring, content assist, and searches. All of these improvements help embedded designers better understand their code and improve productivity.

We've also made a number of enhancements to improve the usability and flexibility of the managed build system. You can now specify pre- and post-build steps, as well as custom steps per file for special file types. You can also specify environment variables and build macros that will be used for the build.

Debug enhancements include user-definable register groups for better organization of registers on the target while debugging. We've also added a new modules view that allows you to inspect the components of the application under debug and to set breakpoints on the contents of those components. Plus, an enhanced memory view improves usability when you're inspecting memory on the target.

Q ECD: Since the release of CDT 3.0 in August, who has adopted it?

(Editor's Note: The Eclipse Foundation press release on CDT 3.0 dated August 22, 2005 mentions QNX, IBM, Intel, MontaVista, Novell SUSE, PalmSource, Tensilica, Texas Instruments, and TimeSys.)

A Schaefer: A number of the contributors to CDT 3.0 are working to adopt it into their products. QNX Software Systems, for example, will integrate CDT 3.0 into the next release of its QNX Momentics development suite. Texas Instruments, meanwhile, is adopting CDT 3.0.x for its upcoming Code Composer Essentials 2.0 for the MSP430 microcontroller. TI is also adopting CDT 3.0.x and beyond for a product roadmap that will address some of its other silicon offerings. Intel is also among the companies integrating the CDT into their products.

Q ECD: What types of features does CDT 3.0 have to improve software development for multicore processors?

A Schaefer: The flexibility and extensibility offered by the CDT offers interesting possibilities for developers working on multicore systems in both symmetric and asymmetric configurations. In Asymmetric MultiProcessing (AMP) systems that involve multiple operating systems and tool chains, a developer can target all of the operating systems from the same code base, using the same IDE. In the upcoming years, we'll likely see multicore AMP systems that run Linux on one core and a real-time operating system on another core. The CDT can integrate with both tool chains and allows for debugging of both cores at the same time.

Q ECD: What types of extensions are developers adding to CDT 3.0 and the managed build system? Can you give a couple of examples?

A Schaefer: A number of vendors are integrating their tool chains into the managed build system. These include Intel and Texas Instruments, who both serve as committers for the managed build system. QNX is also planning an integration in an upcoming QNX Momentics release.

TI is using the new capabilities in the managed build system to provide support for its proprietary code generation tools. This gives its embedded customers the ability to reap the benefits of TI's optimizing compilers – such as global and inter-procedural optimizations for both code size and speed – within an Eclipse environment. TI is also taking advantage of the new capability to customize the New Project wizard to provide a customized project creation workflow for users – this will let users quickly configure their project to properly build and debug for their specific target silicon, embedded operating system, and so on.

There is a lot of interest in the managed build system on the mailing lists and bugzilla, so we anticipate that there are many others, including users, who are working on integrations.

Thanks to the contributions of TI, Intel, Symbian, and others, the managed build system in CDT 3.0 has reached a new level of extensibility for ISVs that we can all reap the benefits of.

ECD: What has surprised you in ways developers are using CDT 3.0?

Schaefer: Certainly the biggest surprise is the great work by the Photran team out of the University of Illinois Urbana Champaign (UIUC) and the Los Alamos National Laboratory, which are taking the CDT and customizing it to work for Fortran. There is still a lot of interest in Fortran for use for complex scientific applications. Right now, the Photran team takes the CDT and, applying a fairly small patch, gives it the Fortran flavor. We look forward to working with this team to make the patch unnecessary and to open the door to further language integrations.

When you look through the list of participants on the CDT newsgroup, mailing list, and bugzilla, you quickly see the diversity of projects that people are building with the CDT. We have folks developing systems from bare-iron embedded systems to large enterprise systems that target mainframes and Linux servers through game consoles. This diversity is what makes the CDT such a rich community and ensures that the CDT remains flexible, extensible, and applicable to a wide array of developers.

We were pleasantly surprised to find out that a developer working on Mozilla has tried the CDT on the Mozilla source base and had a measure of success with it. Mozilla is one of the largest and most complex open source projects, and is the benchmark that the CDT developers use to measure performance and scalability of the CDT. Work continues to ensure the CDT can be used on Mozilla and similarly large-scale projects.

It's also interesting to note that TI uses CDT plus GDB to debug its own GDB ports that support TI silicon.

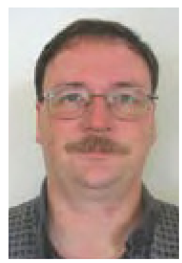
ECD: What's next on the plate for the CDT committers? Is there a CDT 3.1 or 4.0 in the works? Are additional committers stepping up?

Schaefer: CDT will proceed with a six-month release cycle, with additional performance and usability improvements in the works. At October's CDT developer's conference we will discuss future plans for the CDT. All of the contributors to the CDT and others will get together to go through detailed designs and come up with a vision that will take the CDT through the next couple of releases.

We also have additional corporate vendors that are very interested in contributing to the CDT. These topics are on the agenda for the conference in October.

(Editor's Note: This issue of ECD went to press before the conference – please look on www.embedded-computing.com for the latest developments with CDT.)

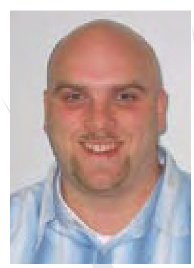
We are definitely excited by the diverse interest in the CDT and will continue to promote contributions that help advance the Eclipse and CDT towards being the defacto-standard IDE for all types of development.



***Doug Schaefer** is project lead for Eclipse CDT and a senior software developer at QNX Software Systems. Previously, he worked for ObjecTime, Rational, and IBM, where he specialized in developing code-generation and software modeling tools. He now brings his technical leadership and experience to the team that develops the QNX Momentics IDE, a development environment based on Eclipse and the CDT.*

To learn more, contact Doug at:

QNX Software Systems
175 Terence Matthews Crescent • Ottawa, Ontario
Canada, K2M 1W8 • Tel: 613-591-0931
Email: DSchaefer@qnx.com • Website: www.qnx.com



***Chris Recoskie** has been with Texas Instruments for the past five and a half years specializing in software development tools. Chris has contributed to the design and development of both the Code Composer Studio and the Eclipse-based Code Composer Essentials integrated development environments. Chris is currently a committer on the build system of Eclipse CDT.*

To learn more, contact Chris at:

Texas Instruments
150 John Street, Suite 801 • Toronto, ON
Canada, M5V 3E3 • Tel: 416-340-2098
Email: crecoskie@ti.com • Website: www.ti.com

Biometrics security through middleware

By Shoieb Yunus

In today's world, businesses realize the equal importance of digital and physical security. Digital data and physical assets are lost – accidentally or deliberately. Biometrics can increase a company's ability to protect its data by implementing a more secure mechanism than a password. In this article, the author discusses device security needs and presents middleware tools to add fingerprint-based security.

A common problem with a password is that it can be compromised easily. Passwords are stolen, forgotten, and shared. Biometrics provides ways to protect data. Sales forecasts, business plans, new product ideas, customer lists, and other critical data can be protected using biometrics.

Biometrics is the only form of security that positively identifies and verifies an individual. Fingerprint, face, iris, and voice recognition technologies are used to determine an individual's identity and their corresponding access privileges. An unauthorized user can fraudulently swipe someone's card or use their password to gain entry into a building or computer, but cannot use their fingerprint or face.

For example, fingerprint readers for door access allow an authorized person's entry into the building. A standalone fingerprint reader enrolls and verifies an individual, but it does not address the issue completely. In a corporate environment, it is imperative to connect this device to the network server. A centralized system can monitor, manage, and verify entry and exit of employees, contractors, vendors, and visitors, even tracking employees' time, attendance, and desktop and network usage as well. In corporate environments, physical security must be tied to network security to be completely effective.

Mobile applications proliferating

Mobile phones are available with much more functionality than the ability to make a simple phone call. More Application Programming Interfaces (APIs) are available for imaging, multimedia, games, and enterprise access. In 2003, more than 84 million units of digital camera phones were sold. A myriad of mobile applications are in development, including video, imaging, document sharing, ERP, CRM, field service, e-mail, SMS, and other applications with access to potentially sensitive data. Some of the applications include secure logon to the device and network, secure access to voicemail, trusted mobile commerce, and many more. There is a stronger need than ever to make these applications secure via a single touch or swipe of a finger.

Various companies including AuthenTec, Fujitsu, and others manufacture high-performance, low-cost fingerprint swipe sensors, such as the AuthenTec EntrePad fingerprint scanner shown in Figure 1, for computers, mobile phones, PDAs, and other devices. However, gaps exist between sensors and mobile devices. A middleware solution could bridge the gap by providing *mobile-aware* software tools for application developers and service providers.

Figure 1

Goals for device security

A good device security system, including biometrics, should provide these functions:

- **Secure logon to devices** – Allow only an authorized user access to a personal computer or other device using biometric or non-biometric (such as smart card) authentication, and record and report on access attempts.
- **Protect application launch** – Allow only an authorized user to start productivity applications (such as accounting, financial, contact management, word processing, CAD, EDA software, databases, and Web browser), and secure those applications against unauthorized access.
- **Encrypt files and folders** – Secure sensitive data via *right clicking* on files/folders or using commands from the application console, and prevent unauthorized users from accessing critical data.
- **Manage password bank** – Securely store passwords for single sign-on to productivity applications and websites, replacing user names and passwords with a convenient, unique authentication.
- **Lock unattended screens** – Screen access (deactivation of *screen saver*) is secured by biometric or non-biometric authentication, protecting the device and data while a registered user is away.
- **Support multiple user authentication methods** – Various combinations of user name, password, biometric, and non-biometric authentication should be supported.
- **Simplify user interface** – Consumers should be able to use security through simple and attractive interfaces, reducing fear and making the technology easy to use without requiring technical knowledge.

EzPassport middleware toolkit

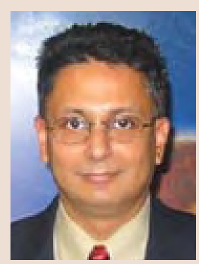
Almost all biometric sensor manufacturers provide a high-level interface or Software Development Kit (SDK), but there is no standard for the high-level functions provided for different sensors, so SDKs from different vendors force system changes if a sensor or data store changes. Also, these SDKs from sensor manufacturers usually cannot be customized easily, so integrators must often deal with a variety of programming tools and interfaces.

The EzPassport family of products is an open application framework, so system integrators or software developers can quickly and easily integrate biometric security features into any Microsoft Windows application. Since the EzPassport middleware is biometric-layer agnostic, it can be integrated with fingerprint, face recognition, iris, retina, voice, signature, and smart cards. It can also be married easily with biometric-enabled fingerprint readers for door entry and other applications.

EzPassport Toolkit, designed in C++, includes API, DLL, header files, and sample code for including EzPassport Plus functionality within any Windows-based software products.

EzPassport Plug-in is COM-based and supports a variety of applications without rebuilding when a new or changed component is used, and can be used very easily with .NET platform and a variety of programming languages such as Visual Basic (VB), C# as well as scripting languages such as JScripting, Java Scripting, or VB Scripting. Because the EzPassport Plug-in is built in Microsoft Visual C, it can also be used by C++ applications.

By using products from the EzPassport family, developers can implement a uniform set of high-level functions from lower-level primitives that understand specific components. For example, if only the functionality of the authentication engine or database engine is required, that component can be accessed by a low-level interface. Developers write applications to a high-level API, minimizing the application changes required when new biometric authentication devices are introduced. **ECD**



Shoieb Yunus *is founder and CEO of EzValidation, Inc., which provides secure, practical, and affordable solutions to the problem of proving positive identity on personal computers, mobile phones, and handheld devices. Prior to founding EzValidation, Shoieb was a marketing consultant at Veridicom, a leading fingerprint technology company, and business development and product marketing manager at Dazzle Multimedia (a subsidiary of SCM Microsystems). He has a BS in Computer Science from the University of Kentucky.*

To learn more, contact Shoieb at:

EzValidation, Inc.
830 Stewart Drive • Sunnyvale, CA 94086
Tel: 408-329-4360 • E-mail: shoieb@ezvalidation.com
Website: www.ezvalidation.com

Creating fast, bit-accurate fixed-point simulation models in MATLAB for accurate verification and prototyping of signal processing algorithms

By Anu Shivaprasad

Traditionally, there have been very few tools and processes for helping migrate from floating to fixed-point math in signal processing algorithms. In this article, the author proposes a systematic method that tracks quantization choices through the model and then tests the resulting quantization selections using the same test bench as the floating-point model. The process is illustrated with modeling of a WLAN (802.11a) algorithm, emphasizing MATLAB methods for accelerating simulation, adding quantization information, analyzing results, and modifying the quantization to achieve acceptable behavior.

Many of the complex mathematical algorithms used in digital signal processing applications are developed in floating-point math because it is easier to develop and debug the algorithm without worrying about quantization effects. However, when these algorithms are deployed in a target system – on DSPs, FPGAs, or ASICs – they will most usually be represented in fixed-point math. The fastest, most power-efficient, and least expensive DSPs typically employ fixed-point math, while fixed-point math is mandatory when an algorithm is implemented on an FPGA or ASIC. Going from floating to fixed-point math is an important step in the design process. Making the wrong quantization selection can have significant, even disastrous, effects on the behavior of the algorithm.

Figure 1 shows a block diagram of the example WLAN model developed in MATLAB. All blocks represent individual MATLAB modules. By default, all MATLAB modules are represented in floating-point. The first task in the floating-to-fixed-point process is to create a floating-point model optimized for simulation speed since a large number of simulations will be run. Some of the techniques used include:

- Separating the test bench and file I/O from the computational code
- Avoiding dynamic array extensions
- Initializing arrays where possible
- Using integer variables where possible

These techniques will not only help accelerate the execution of the standard interpreted MATLAB, but will also allow compiling the MATLAB model and accelerating the simulation. Accelerated simulation will be particularly valuable when going from floating-point representation to fixed-point representation as fixed-point simulations usually take orders of magnitude longer than the floating-point counterpart. This is because modeling the fixed-point arithmetic is not native to the host processor. For example, simulating saturation arithmetic requires modeling the saturation logic used in the final hardware implementation, which significantly slows the execution. But with acceleration techniques, the execution speed of the fixed-point can be improved, making it fast enough for executing large test suites.

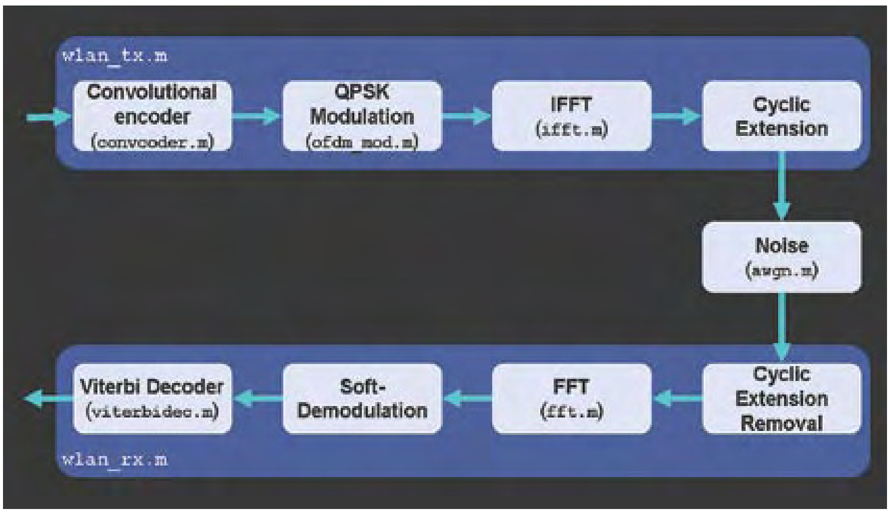


Figure 1

Linked C code accelerates model

The first step in modeling the WLAN algorithm is to run Bit Error Rate (BER) and Frame Error Rate (FER) simulations on the floating-point model for various Signal to Noise Ratio (SNR) conditions. The results are indicated in Figure 2. After verifying that the model behaves in the expected manner, the simulation can be accelerated by generating a MEX file of the model. A MEX file can be created by writing the C-code equivalent of the MATLAB function at any level of the system hierarchy and then linking it in with the existing MATLAB model through the standard MATLAB API. C-code equivalents of MATLAB functions are frequently written by hand to speed execution of time-consuming modules.

Catalytic RMS can automatically generate MEX files, which was done in this case for the entire model. The accelerated WLAN model executes in 0.6027 seconds, for an improvement factor of 500X over the original interpreted floating-point model. Accelerating the simulation allows execution of more simulations with the obvious benefits of either allowing more tests to be performed, performing the same amount of testing in a shorter time, or some combination of the two. But going through the process of automatically generating a MEX file has another benefit: it also guarantees that fixed-point modeling information can now be added into the model, the model can further be analyzed to determine the quality of the quantization choices, and the quantized model can then be accelerated for faster simulation.

Sometimes quantization choices are constrained by the hardware targeted for implementation. For example, if the algorithm is being implemented on a standard DSP, the input data sizes and internal data structures will be set to multiples of 8 bits, typically

16 or 32 bits. But if the algorithm is being implemented on an FPGA or as an ASIC, the data sizes should be represented in the fewest number of bits needed to maintain the numerical integrity of the algorithm, thereby minimizing the silicon required to implement the algorithm. For both hardware and software implementations it is frequently desirable to analyze the quantization options.

When adding quantization for fixed-point representation, the first step is to analyze the model and gather information about the behavior of the floating-point model under simulation. One method for gathering quantization information is to capture every value that each variable assumes during the simulation to determine their numerical range. Selecting the *profile* option when automatically generating a MEX file using the Catalytic RMS tool creates an instrumented version of the MEX library that, when executed, collects the range of values for each variable during the simulation. This information can be graphically viewed using the Catalytic Value Profiler as shown in Figure 3. The histogram represents the distribution of base-2 logarithm of the absolute value of the variables, with each bin representing one bit of precision. It further suggests a fixed-point format to quantize any variable to avoid overflows.

The goal of using the Value Profiler tool is twofold:

- To provide a starting point for quantization. Since the range information is derived from simulations it is important to choose representative input data.
- Gathering information on quantized models to highlight any problems, such as overflow.

Based on the floating-point profile information and the choice of target hardware, quantization can be determined. In this case, the target processor is the Texas Instruments TMS320C6416, and variables with word lengths that are multiples of eight are chosen for quantization. As shown in Figure 3, the profiler recommends that the variable metric be quantized with 13 integer bits. A standard word width is 16 bits and, from the profile information, 13 integer bits are needed to represent the MSB to cover the range of the data. What is undetermined is if 3 fractional bits (16 - 13 bits) will adequately represent the precision needed. The fixed-point representation is added to the model using Catalytic fixed-point constructs, and Catalytic RMS propagates the quantization information through the

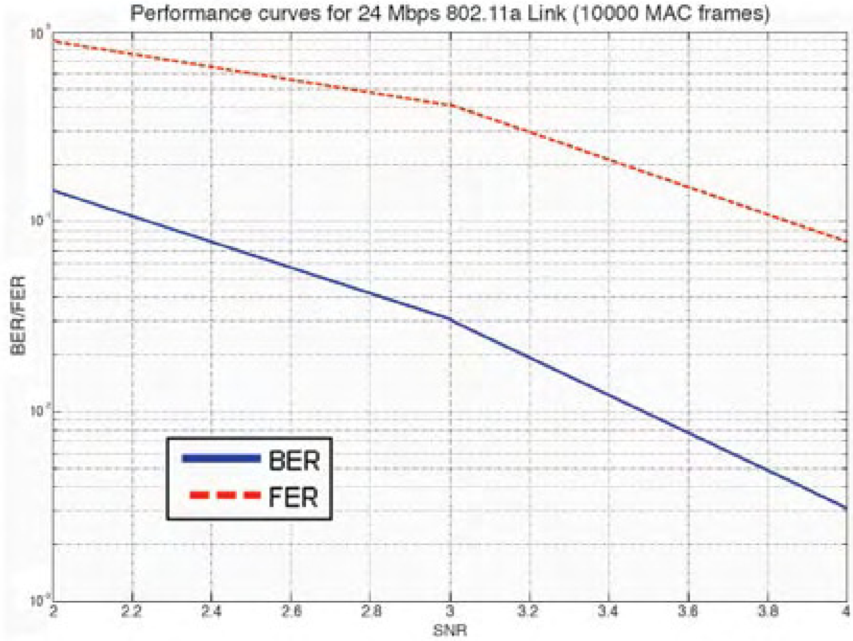


Figure 2

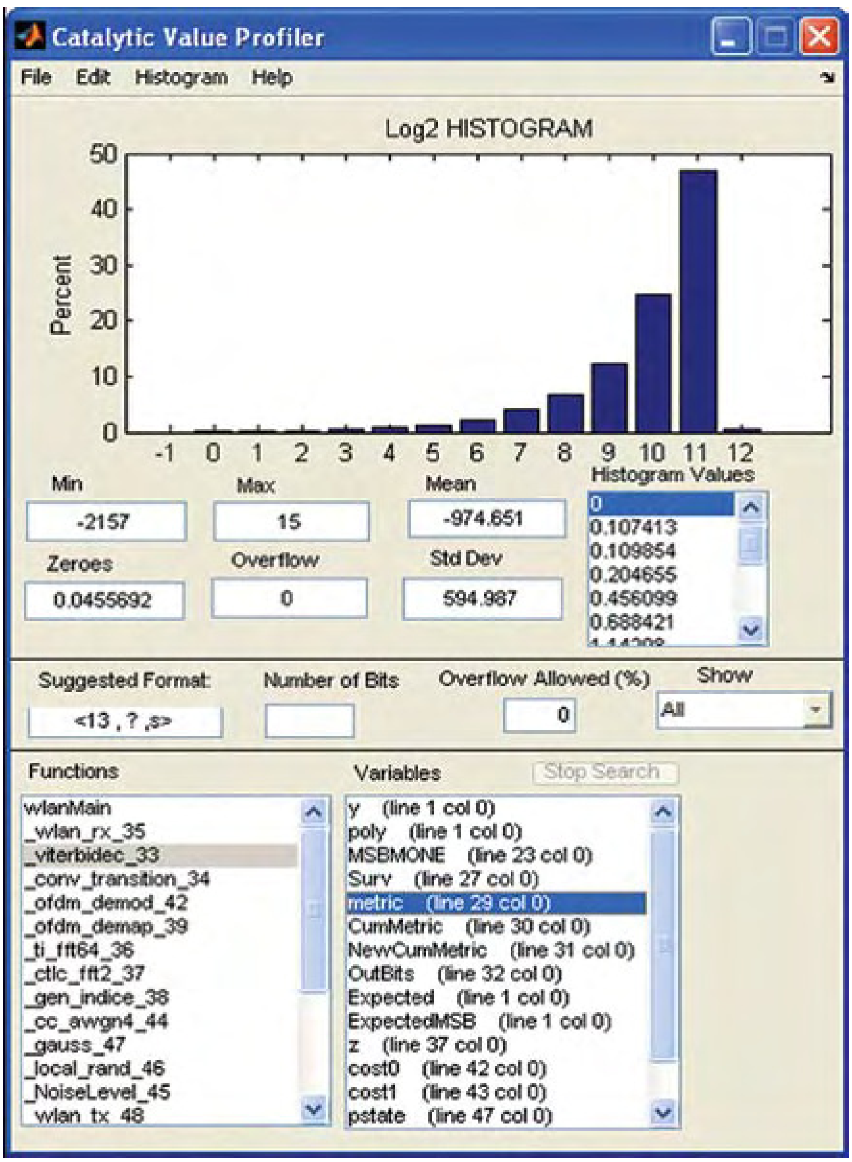


Figure 3

WLAN in preparation for running the same simulation with fixed-point math.

Figure 4 shows the Catalytic RMS Project Management screen with the propagated fixed-point types indicated in the right-hand pane. Then the quantized model is simulated to see if the precision is adequate.

With quantization chosen, the quantized model is simulated to see if the precision is adequate. Figure 5 shows the same BER and FER plots rerun in the quantized fixed-point model and compared to the floating-point results. To create these charts, fixed-point constructs are added to the MATLAB source (as shown in Figure 4), a MEX file is generated to speed the simulation, and then the results plotted on the same graph as the floating-point results. The fixed-point results diverge slightly from the floating-point results, which are expected. The quantization can be further refined by manipulating the fixed-point constructs in the MATLAB model or by testing on the target hardware. If this design were to be implemented, further refinement of the quantization choices would probably be performed.

Not all algorithms perform equally well

Consider what happens when a significant modification to the algorithm is made. Initially, only the QPSK demodulation scheme was employed in the Soft Demodulation block, and with the selected quantization the algorithm performs well. However, perhaps a QAM-16 demodulation scheme is desired. After adding the QAM-16 function and doing initial testing in interpreted MATLAB, a new MEX file is generated and extensive tests in floating-point are run. Once satisfied with the floating-point behavior, the same quantization steps are done, another MEX file is generated, and fixed-point simulation is rerun.

As shown in Figure 6, the simulation results of the modified WLAN with the QAM-16 functionality are unacceptable. Given that the floating-point simulation exhibited correct behavior it is suspected that the quantization choices are poor for this algorithm. To investigate, once again an instrumented MEX file is generated, and the Value Profiler is used to help determine if there is an overflow in the variable metric.

Once the quantization problem has been determined, choose a new quantization by setting the metric variable to 16.0 and rerun the simulation. The quantized model now more closely matches the floating-point models as shown in Figure 7.

Conclusion

A MATLAB model of a 802.11 WLAN was used to illustrate a design problem frequently encountered when taking a floating-point algorithm to an implementation platform. For complex signal-processing algorithms, engineers frequently create floating-point reference models in MATLAB. These algorithms will eventually be deployed in a DSP,

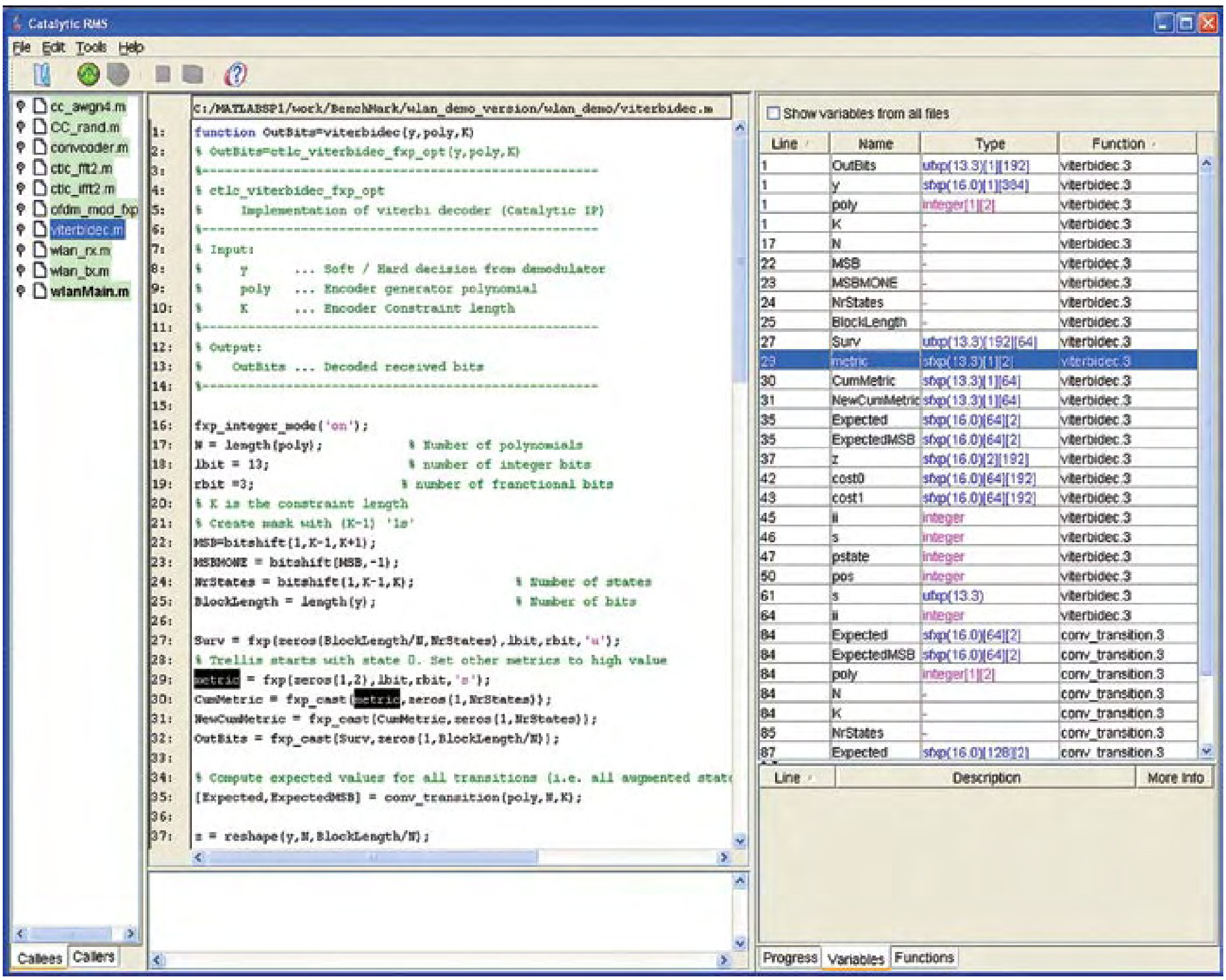


Figure 4

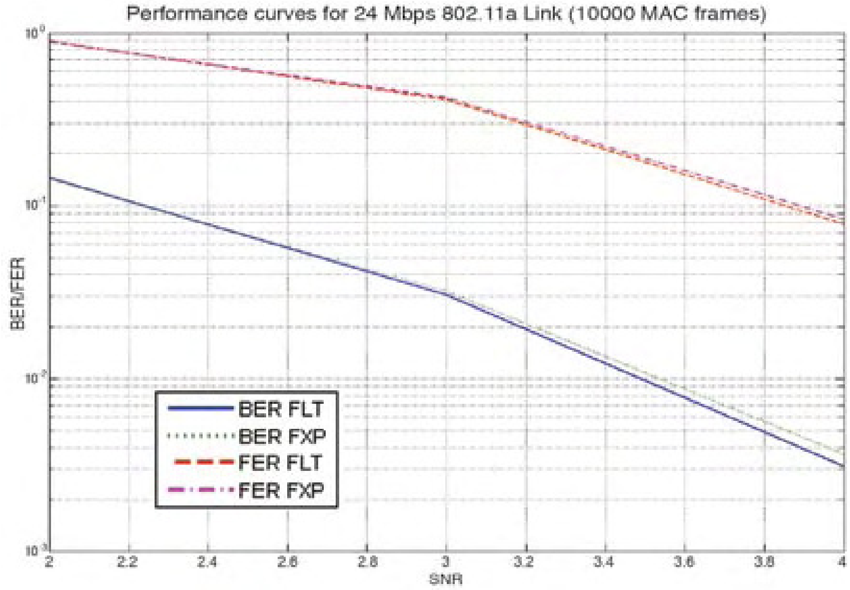


Figure 5

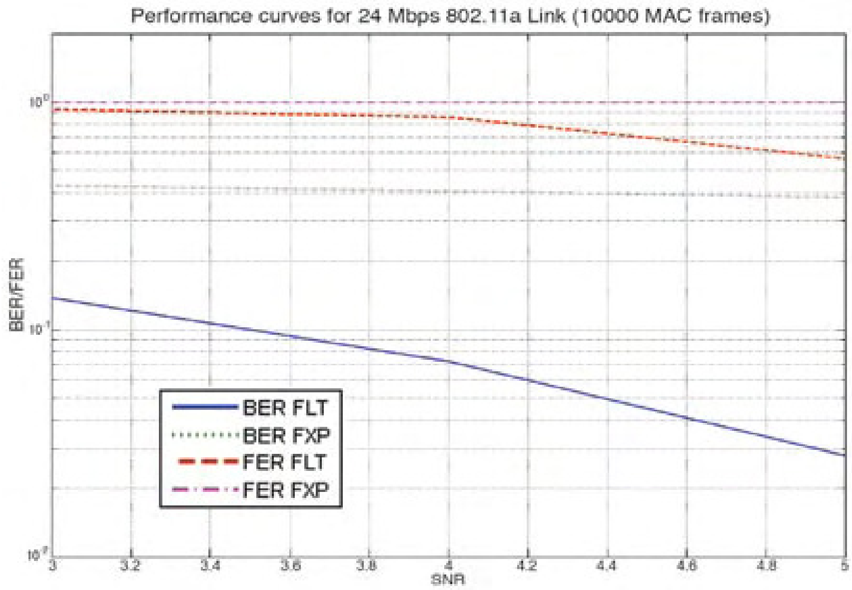


Figure 6

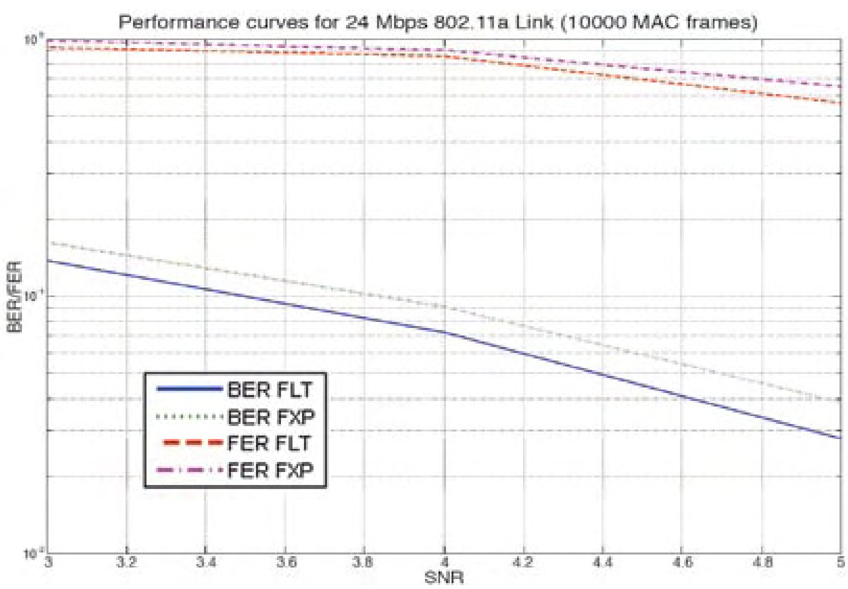


Figure 7

FPGA, or ASIC implementation. The implementation often is done in fixed-point to meet design constraints, including real-time performance demands, system cost, and power consumption.

The method presented is in contrast to the more traditional design flow as indicated in Figure 8. Traditionally, after verifying the accuracy of the floating-point reference model in MATLAB, the next step has been to hand-translate the MATLAB to floating-point C,

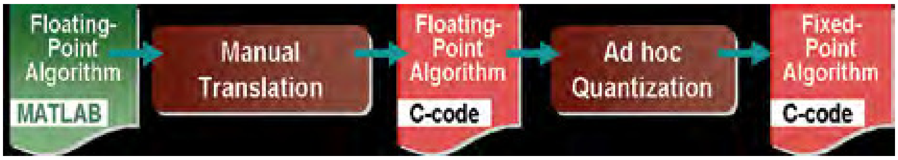


Figure 8

> "The new method proposed not only speeds the simulation of the algorithm in the MATLAB environment, but also enables fixed-point exploration in the MATLAB environment..."

compare the accuracy of results with the original MATLAB model, and add fixed-point data types to the C-code.

Using the traditional process, the ability to analyze the fixed-point characteristics in the superior MATLAB environment is lost. It is also easy for the original MATLAB and the fixed-point C-code version to diverge since there is no automatic way to go from floating-point MATLAB to fixed-point C-code.

The new method proposed not only speeds the simulation of the algorithm in the MATLAB environment, but also enables fixed-point exploration in the MATLAB environment and allows the fixed-point C-code to be systematically derived from the floating-point MATLAB. **ECD**

***Anu Shivaprasad** is a staff applications engineer at Catalytic Inc. She is currently working on products that facilitate the fast implementation of MATLAB algorithms on signal processing platforms. Her background includes signal processing, image processing, and communication algorithms as well as detailed knowledge of MATLAB.*

To learn more, contact Anu at:

Catalytic Inc.
1076A East Meadow Circle
Palo Alto, CA 94303
Tel: 650-846-2555 • Fax: 650-846-2557
E-mail: anu@catalyticinc.com
Website: www.catalyticinc.com

Extending Windows XP into real time

By Paul Fischer

The use of Windows in the embedded marketplace continues to grow, acknowledged directly by Microsoft's aggressive promotion of two Windows platforms for the embedded market: Windows CE and Windows XP Embedded (XPe). CE addresses the small and mobile marketplace, whereas XPe is a specially packaged version of Windows XP with some features intended for complex embedded applications. This article looks at the impact of dual and multicore microprocessors with a combined Windows XP and Real-Time Operating System environment.

Standard Windows XP is a rich platform on which to build complex applications. The familiar user interface allows one to build embedded instruments that can be quickly mastered by the end-user. Modern PC-based computers can display complex images and interface with a wide variety of networking protocols.

Windows XP Embedded (XPe) adds several key features that are crucial to OEMs for the use of Windows in an embedded application:

- Simpler licensing terms than standard Windows
- Custom configuration of Windows (by removal of unnecessary components)
- *Hibernate once, resume many* and avoidance of *first time install* messages
- *Gold disk* manufacturing process for more efficient factory floor production techniques

Avoiding *first time install* messages, the ability to build a fully configured system at the factory floor with a single unique image, and removal of unused Windows features are all key reasons for choosing XPe. Even though a system is based on Windows, it is not required that the entire Windows environment be exposed to the end-user. Windows XPe accommodates removal of those parts of the system that are either unnecessary or, by their inclusion, might compromise the usability or safety of the system. In other words, XPe provides a means to constrain the embedded Windows device in order to operate strictly for the function(s) for which it was intended.

XPe's unique *hibernate once, resume many* feature insures a consistent initial system state, even following an unplanned system shutdown (such as pulling the plug). Startup is rapid and can commence with the primary device application loaded and ready for use, taking less than 10 seconds (following completion of the BIOS POST) in a typical system.

Real-time software requirements

A *real-time system* is one in which the time to respond to an event is as important as the logical correctness of the response. Typical hard real-time systems require a level of determinism and performance that can be measured in the tens of microseconds.

Windows XP Embedded is a tool for developing a managed OS feature set. This is critical for applications in which the components of the operating system must be known and controlled. Despite all of its advantages, the XPe tools do not address determinism. The number of processes allowed to run can be limited by XPe, but the system's overall lack of real-time determinism cannot be fixed.

Windows XP and XPe are capable of providing fast overall response to events; however, neither is appropriate on its own for applications that require true hard real-time determinism. A Windows application thread, regardless of its priority, can be preempted by one of many software and hardware sources – including interrupts and high-priority kernel and driver threads.

Real-time software must also be robust and reliable. While a programming error in a business application might lower the productivity of the user, an error in a real-time control program can result in costly downtime, damage to expensive equipment, or even the loss of human life. Tools and protection mechanisms must be made available to the real-time developer to aid in minimizing the occurrence of typical programming errors such as stray pointers, memory leaks, and uninitialized variables. In the event of faulty code and/or a software crash, protection mechanisms minimize the impact of a software crash on the critical processes being controlled.

Why Windows XP is not real-time

Windows is built around a fully preemptive, multitasking kernel that is capable of satisfying, on average, an application's timing and response requirements. This means that Windows will miss some scheduling deadlines, and will be late for many other deadlines. Whether or not this condition is acceptable depends upon the requirements of the application.

Windows will miss or be late for deadlines due to some fundamental kernel policies. These policies were put in place for a very good reason: to optimize the average system performance by being fair to all applications on the machine. These policies spell trouble for the real-time systems programmer; real-time applications are not fair, and a real-time application must be selfish with regard to its use of the processor when compute cycles are needed.

Determinism (the ability to meet deadlines predictably) is what separates real-time applications from general-purpose applications. A deterministic application can only be built if the timing of events managed by the operating system, upon which it relies, is reliable and predictable and the developer is allowed extensive control over the relative priorities of all operations and events. Windows restricts the ability to control and predict threads and events in a number of ways:

- Because the Windows priority spectrum places all interrupts at a higher level than normal thread execution, user-level threads will always be preempted by any interrupt source, regardless of the importance of that interrupt. This means that even moving a mouse will generate an interrupt that preempts a high priority non-interrupt operation. Only kernel-level threads can raise or lower the Interrupt Request Level (IRQL) to mask or unmask interrupts. In configurable Real-Time Operating Systems (RTOS), thread priorities can be interleaved with interrupt priorities, giving the developer total control over the relationship between interrupts and threads.
- Windows utilizes a Delayed Procedure Call (DPC) mechanism to increase the responsiveness of the system to interrupts. A correctly designed Interrupt Service Routine (ISR) minimizes interrupt latency by performing only critical processing in the ISR and then queuing a DPC for further processing as a thread. DPCs are placed into a single FIFO, with no provisions for the priority of the operation. This means that a low priority DPC will execute first, regardless of the priority of other DPCs queued behind it. You can cheat and place a DPC at the head of the queue, but this does not solve the problem because you may inadvertently be deferring the execution of another higher priority DPC that is already in the FIFO. Additionally, since DPCs are lower on the priority spectrum than all other types of interrupts, DPCs will not execute before interrupts – even low priority interrupts.
- Multiple requests for a synchronization object in Windows (such as a semaphore or a mutex) are also queued in a FIFO, again without regard for the priority of the requesting thread. Thus a high priority thread may have to wait for a low priority thread to complete its operation before proceeding. This affects not only determinism, but can also lead to priority inversion.
- Solving the classic problem of priority inversion requires the ability to inherit the priority of another thread, which is not available in Windows (Windows uses a random boosting mechanism that does not always solve the problem in adequate

time). The inversion problem can be described as follows: assume at least three threads, A, B, and C, with A being the highest priority and C the lowest. Priority inversion occurs if C has previously locked a resource and A is waiting on that resource, but C is unable to complete its job because it is has been preempted by B. Thread A has been effectively held off by lower priority thread B. Temporarily boosting C's priority to A's (C inherits A's priority level) remedies the problem.

Virtual hardware via software

To balance the flexibility of Windows with the deterministic requirements of embedded applications, designers usually add a dedicated real-time component (in effect, a second computer). This *dual-computer dual-OS* solution is built upon two distinct hardware computing elements: one computer system to run Windows and a second system to host a dedicated RTOS.

Unfortunately, a second control computer adds substantial cost of goods, manufacturing complexity, and system-to-system coordination headaches. A *single-computer dual-OS* system, where one compute element hosts both the Windows system and the RTOS, significantly reduces the cost of goods and complexity, and simplifies the coordination of Windows with real-time processes. It can also reduce design costs, measured in the time and effort spent on engineering tools and staff, which alone may be sufficient reason to seek this simpler solution.

This might seem like an unrealistic proposition, but it is possible for one hardware platform to support two operating systems using virtual machine technology. By running the RTOS in one of the virtual machines, a *dual-computer dual-OS* architecture is converted into a *single-computer dual-OS* solution. However, in order for such a solution to be considered a viable option it must, at the very least, satisfy the following criteria:

- The RTOS must be safe, secure, reliable, and extensible
- The hardware platform must support dependable, hard real-time functionality
- The application must be easy to design, build, and integrate using standard tools

Applications built for a *single-computer dual-OS* system must be partitioned into deterministic and nondeterministic parts. The nondeterministic application executes on the Windows virtual machine, and the deterministic application executes on the real-time virtual machine. The two application parts must have managed access to a variety of shared objects (such as mailboxes, semaphores, mutexes, shared-memory) to work together properly.

Real-time processes and threads running on the RTOS virtual machine need access to high-speed interval timers for accurate, low-drift time measurements and for generating exact periodic intervals to insure precise control of real-time systems. x86 Advanced Programmable Interrupt Controller (APIC) uniprocessor systems and multi-processor systems, the vast majority of embedded and desktop Windows

"The ability to use a single standard development tool for both environments is a significant advantage of the virtual machine."

platforms built today, are excellent candidates. The accuracy and drift of the timer elements in x86 APIC systems is very good.

The ability to use a single standard development tool for both environments is a significant advantage of the virtual machine. The real-time environment needs direct access to I/O and memory, a fixed priority scheduling system with priority-inversion protection, and simplified interrupt-handling services to insure efficient implementation of real-time threads. This allows developers to create and deploy sophisticated real-time applications without having to write complex and cumbersome device drivers for Windows XP for access to real-time hardware.

The virtual machine *single-computer dual-OS* approach eliminates redundant computer and communication hardware and improved communication and coordination between the real-time and Windows applications.

Dual-core enhancements

The process described previously, using two virtual machines to share a single CPU platform that supports Windows and real time, works for a large number of real-time Windows applications. Typically, applications with cycle times of one millisecond or slower are served quite well by this arrangement and have been deployed on the current crop of desktop and industrial motherboard platforms (uniprocessor and hyper-threaded Pentium 4 class processors running at 1-3 GHz). There are, however, some applications that demand faster cycle times. For these applications there is another solution that is just now possible with the introduction of dual-core processors into the mainstream.

This alternate solution retains the cost and efficiency benefits of the virtual machine *single-computer dual-OS* approach, but no longer requires sharing the CPU and key processor resources. When virtual machines share a CPU, as is the case with the single-core processor designs common today, they must maintain a full machine context (or a partial context in the case of a hyper-threaded core) in order to switch between the two operating systems. Saving and restoring these contexts results in some compromises regarding event response latencies and maximum cycle times. These compromises typically contribute on the order of 10 to 30 microseconds to interrupt jitter. For cycle times of one millisecond or slower 10-30 microseconds of interrupt latency represents a jitter variation of only a few percent.

Higher speed cycle times mean higher bandwidth controllers, a desirable trait because it leads to improved performance and throughput. However, 10-30 microseconds of jitter is a significant number when cycle times of 50-200 microseconds are needed. If jitter is a significant percentage of the cycle time it adversely affects the quality of the control algorithm. The stability and quality of the controller are a function of the accuracy of the cycle time. Variations in cycle times degrade the stability margin of a closed-loop control system, especially naturally unstable systems like position-feedback motion control loops.

Dual-core processors can support two operating systems by dedicating one CPU to the RTOS. The CPU instruction cycles of the dedicated core are available 100 percent of the time to the RTOS. The CPU cycles of all remaining cores become the exclusive property of the Windows virtual machine. Contention for key CPU resources such as pipelines, cache, and the FPU are avoided. Using the built-in interprocessor communication mechanisms accomplishes coordination between the two processors, eliminating context switch times entirely. In this scenario, interrupt latencies are reduced by an order of magnitude, from 10-30 microseconds down to 1-3 microseconds. Loop cycle times in the 50-200 microsecond range can operate with high precision and accuracy. The advent of inexpensive dual-core hardware means an order of magnitude in quality and bandwidth control algorithms can be implemented on a real-time Windows platform.

In addition to vastly decreased jitter, a dedicated RTOS processor in a dual-core system has the advantage of 100 percent dedicated CPU cycles. When two operating systems share a single CPU they must be willing to share the raw compute performance in order to assure that both operating systems will be able to execute and perform the tasks at hand. In other words, if one OS consumes all the CPU cycles the other OS will effectively be *frozen out*. When a CPU can be dedicated to each OS this is no longer a concern. On a dual-core system the real-time tasks can maximize the number of compute cycles they consume (because they can consume 100 percent of their CPU's cycles and Windows will still run). This means that even more complex control algorithms can be developed, resulting in further increases in the quality and performance of the control systems that can be implemented on real-time Windows systems.

Safety and reliability by design

Note that the virtual machine approach to adding real time to Windows is quite different from installing a real-time kernel in the form of a Windows device driver or subsystem. The device driver and subsystem models force real-time applications to operate as part of the Windows kernel. Kernel mode code has privileged access to the entire memory space, including the Windows kernel and other device drivers; it lacks address isolation and memory protection. A real-time thread running on such a system can easily

overwrite other processes, both real-time and Windows processes. Because such programming errors are difficult to detect in kernel mode and result in spurious but critical failures, achieving reliable operation through this method often requires extensive testing and debugging, with many errors not detected until the system has been deployed in the field. Creating a complex, multi-threaded, real-time application to run inside the Windows kernel is contrary to the notion of building reliable, safe, and dependable real-time applications.

By definition, when Windows crashes (for example, blue screen) something catastrophic has occurred and Windows itself cannot recover. The integrity of all of Windows is in question, including interrupt handling, the kernel, and device drivers. Thus, continued operation of a real-time driver or subsystem that is encapsulated within the Windows kernel, following a Windows crash, is unreliable at best, and will likely result in failure of the real-time subsystem and all of its processes.

Using a virtual machine to add real time to Windows means the RTOS will maintain reliable operation of real-time processes in the event of a Windows crash. The virtual machine approach to real-time Windows allows real-time applications to run in user-mode, not kernel-mode. The result is improved reliability and robustness, as well as simplified programming and debugging. Each real-time process built on a virtual RTOS runs in a separate 32-bit protected memory segment. These segments are distinct from those used by Windows and provide address isolation and protection not just between the real-time processes, but also between real-time processes and non-real-time Windows code.

Real implementation – INtime for Windows

TenAsys Corporation has implemented such a system: the INtime RTOS running in a virtual machine alongside Windows. The INtime RTOS virtual machine makes it possible to extend Windows applications into the real-time domain by providing a separate hard real-time virtual machine on which the real-time components of an application reside. A complete real-time Windows application consists of both non-real-time Windows processes and threads, and real-time processes and threads. Real-time processes typically handle time-critical data acquisition and control, while non-real-time processes handle the human interface, network communications, and data storage.

The complete *single-computer dual-OS* system consists of the following key components, as illustrated by Figure 1.

Standard Windows XP

Because the system is divided into two virtual machines, there is no need to utilize a special version of Windows; a standard Windows distribution can be utilized (such as Windows XP, XPe, Windows 2000). Non-real-time processes and threads execute normally on the Windows kernel. Off-the-shelf applications can be used without change; they are unaware of the real-time kernel.

Real-time kernel and API

The real-time kernel provides deterministic scheduling and execution for real-time processes and threads. Real-time interrupts and threads preempt the execution of all Windows threads and disable non-real-time interrupts. Real-time applications utilize the real-time API to access the capabilities of the real-time kernel.

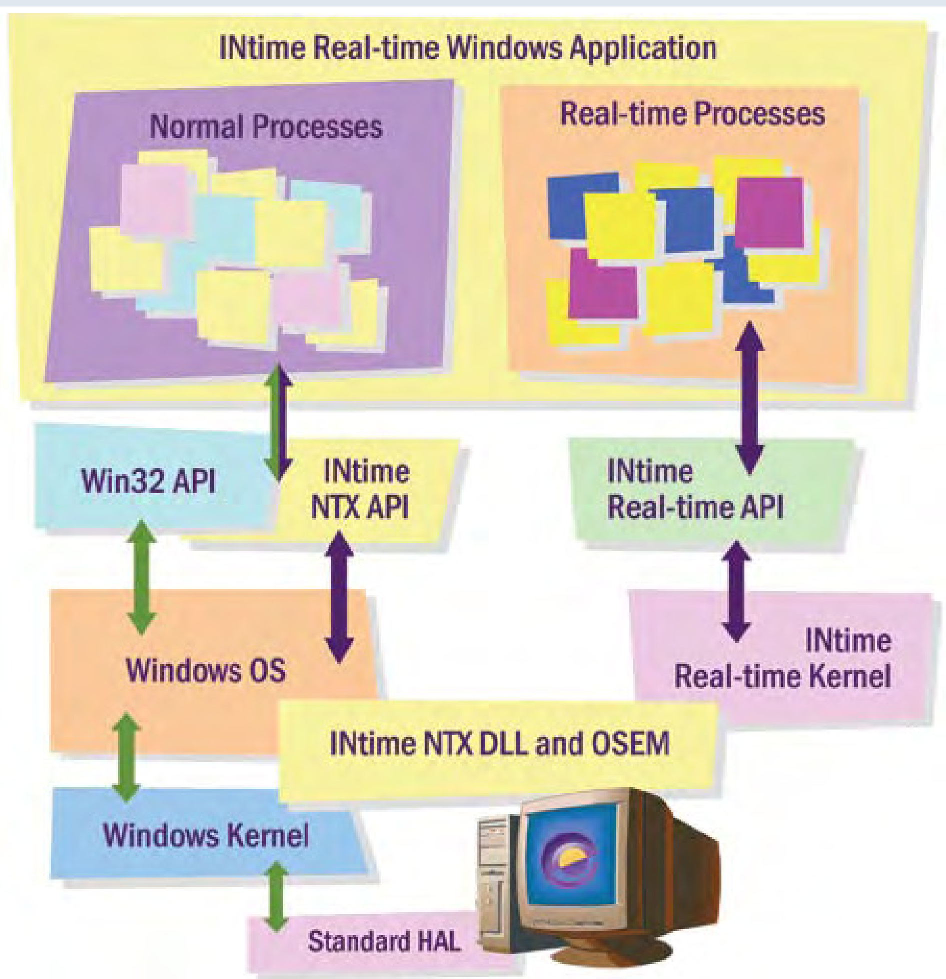


Figure 1

NTX, API, and DLL

The NTX interface provides a mechanism for communication between the Windows virtual machine and the INtime virtual machine. NTX provides applications with managed access to a variety of shared objects, such as mailboxes, semaphores, mutexes, and shared-memory.

OS Encapsulation Mechanism (OSEM)

The INtime OSEM manages the virtual machines to insure simultaneous operation and integrity of the Windows kernel and the real-time kernel. This virtual machine mechanism provides memory protection and address isolation between all Windows processes and real-time processes.

Shared development environment

Because the system includes Windows, it is not necessary to have a separate development workstation; the target system is also the development system. Standard Windows development tools, such as the Microsoft Visual Studio tools, can be used for both the Windows part and the real-time part of an application. A single compiler, linker, editor, and debugger serve both virtual machines.

Conclusion

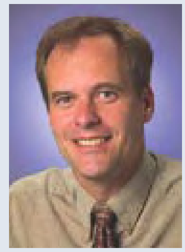
When applying Windows XP to time-critical applications, a real-time operating system is necessary to satisfy the requirements for accurate and repeatable data acquisition and control. An RTOS that shares the CPU with Windows, using virtual machine technology, allows embedded Windows applications to take full advantage of the Windows' standard user interface, network capabilities, development tools, and off-the-shelf software and still deliver the performance required by critical, hard real-time tasks.

Because the virtual machine approach to integrating an RTOS with Windows does not require special hardware, it does not result in increased hardware complexity. Software development is also simplified because the real-time developer can use the same Microsoft Visual Studio development environment used to build standard Windows XP applications.

The virtual machine architecture insures that Windows-based real-time systems always perform data acquisition and machine control at the highest priority; with overall supervisory control and display of data on the user interface running at the lowest priority. Real-time events always run at the highest priority level, regardless of the version of Windows or the hardware platform used.

TenAsys Corporation provides INtime Real Time Extensions for Windows, a fully featured RTOS derived from iRMX. Applications written for the INtime RTOS execute with guaranteed determinism as fully protected user-mode processes in concurrence with the Microsoft Windows operating system on standard PC hardware. Because real-time application code executes in user-mode, the INtime environment is immune to application faults that crash kernel-mode driver solutions.

Using the INtime real-time operating system extension for Windows, designers can reduce the cost of implementing computer-based control systems by utilizing x86-based PC hardware and Microsoft Windows software. INtime software combines the benefits of the Windows operating system – standard APIs, networking, user interface, and development environment – with a proven, highly reliable, real-time kernel designed for critical applications. ECD



Paul Fischer *is a senior technical marketing engineer at TenAsys Corporation. Paul's experience with INtime goes back to 1997, when the product was first introduced. He has more than 20 years experience with real time and embedded systems in a variety of engineering and marketing roles. Fischer has an MSE from UC Berkeley and a BSME from the University of Minnesota.*

To learn more, contact Paul at:

TenAsys Corporation
1600 NW Compton Drive, Suite 104
Beaverton, OR 97006
Tel: 503-748-4720 • Fax: 503-748-4730
E-mail: info@tenasys.com
Website: www.tenasys.com

Windows CE 5.0 for real-time systems

Although Windows CE exposes the same Win32 Application Programming Interfaces (APIs) present on Microsoft desktop and server operating systems, its underlying operating system architecture is completely different from its desktop cousins. Windows CE – designed from the ground up to be a small footprint, componentized, hard real-time embedded operating system – combines support for desktop application development frameworks including Win32, MFC, AT, and .NET with a real-time kernel providing the operating system primitives needed to support today's real-time embedded system designs. This article focuses on the operating system tools used to verify Windows CE 5.0 real-time behaviors.

By Mike Hall

It is understood that real-time systems are not tested with a single analysis that pronounces them correct. Testing of real-time systems is a proof by exhaustion. Developers work to gain confidence in the solution. Microsoft tools work together to further the developer's understanding of a real-time system by providing complete timing explanations of the application and operating system interactions.

There is much discussion about the definition of real time, so it's probably good to take a look at that definition. A quote from comp.realtime FAQ, by Donald Gilles at the University of British Columbia, gives the canonical definition of real time as follows:

"A real-time system is one in which the correctness of the computations not only depends upon the logical correctness of the computation, but also upon the time at which the result is produced. If the timing constraints of the system are not met, system failure is said to have occurred."

CE fits 95 percent of real-time needs

Since the industrial automation applications of high speed I/O, robotics, and machinery controls often create the most demanding timing constraints, Microsoft chose to ask that community for their specific requirements for a real-time embedded operating system. General Motors Powertrain Group (GMPTG) has been the leader in implementing Open Modular Architecture Controls (OMAC) technologies in its manufacturing applications since 1986, and GMPTG later drove the formation of an OMAC User Group. OMAC reviewed hundreds of applications and found that 95 percent of systems require cycle times of one millisecond and greater. A tolerable variation on this one millisecond cycle time would be 10 percent, or 100 microseconds (µs). With a response time averaging 50 µs on a 200 MHz x86 system, Windows CE meets or exceeds the requirements of all but 5 percent of applications OMAC reviewed.

Large portions of industrial automation applications reviewed were driven by an external signal from a piece of machinery. This signal is presented to hard real-time applications in the form of an interrupt. Microsoft has been encouraging Windows CE developers to put as much application code into the Interrupt Service Threads (ISTs) as possible. This leads the OMAC jitter definition to become a timing constraint on IST latencies of no more than 100 µs. Windows CE typically provides latencies in the sub 10 µs range. The remainder of the applications reviewed use timers to create their cycle times. This led to the requirement of 1 millisecond timers with no more than 100 µs latency or jitter. In summary, the OMAC definition provided the following design and test requirements:

- IST latencies of no more than 100 µs latency
- 1 millisecond timers with maximum of 100 µs latency

It is also important to distinguish between a real-time system and a real-time operating system. A real-time system contains all elements, including hardware, operating system, and applications. A real-time operating system is just one element of the complete real-time system. For more information, take a look at *Designing and*

Optimizing Microsoft Windows CE for Real-Time Performance by Microsoft.

Servicing interrupts

A key feature of kernel real-time performance is the ability to service an interrupt within a specified amount of time. Interrupt latency refers primarily to the software interrupt handling latencies – that is, the amount of time that elapses from the time that an external interrupt arrives at the processor until the time that the interrupt processing begins.

Windows CE interrupt latency times are bounded for threads locked in memory, if paging does not occur. This makes it possible to calculate the worst-case latencies – the total times to the start of the ISR and to the start of the IST. The total amount of time until the interrupt is handled can then be determined by calculating the amount of time needed within the ISR and IST.

ISR latency

ISR latency is the time from when an IRQ is set at the CPU to when the ISR begins to run. Three time-related variables affect the start of an ISR:

- A = Maximum time that interrupts are off in the kernel. The kernel seldom turns off interrupts, but when they are turned off, it is for a bounded amount of time.
- B = Time between when the kernel dispatches an interrupt and when an ISR is actually invoked. The kernel uses this time to determine what ISR to run and to save any register that must be saved before proceeding.
- C = Time between when an ISR returns to the kernel and the kernel actually stops processing the interrupt. This is the time when the kernel completes the ISR operation by restoring any states, such as registers, that were saved before the ISR was invoked.

The start of the ISR being measured can be calculated based on the current status of other interrupts in the system. If an interrupt is in progress, calculating the start of the new ISR to be measured must account for two factors: the number of higher-priority interrupts that will occur after the interrupt of interest has occurred and the amount of time spent executing an ISR.

Both Windows CE and OEM code affect the time to execute an ISR. Windows CE and OEM code combined are in control of the variables A, B, and C, all of which are bounded.

IST latency

IST latency is the time from when an ISR finishes execution – that is, signals a thread – to when the IST begins execution. Four time-related variables affect the start of an IST:

- B = Time between when the kernel dispatches an interrupt and when an ISR is actually invoked. The kernel uses this time to determine what ISR to run and to save any register that must be saved before proceeding.
- C = Time between when ISR returns to the kernel and the kernel actually stops processing the interrupt. This is the time when the kernel completes the ISR operation by restoring any state, such as registers, that were saved before the ISR was invoked.
- L = Maximum time in a kernel call (Kcall).
- M = Time to schedule a thread.

The start time of the highest-priority IST begins after the ISR returns to the kernel and the kernel performs some work to begin the execution of the IST. The IST start time is affected by the total time in all ISRs after the ISR returns and signals IST to run. The resulting start time is as follows:

Start of highest priority IST =

$$C + L + M + \sum_{x = 0}^{N_{ISR}} (E + C + T_{ISR_N})$$

Both Windows CE and the OEM affect the time required to execute an IST. Windows CE is in control of the variables B, C, L, and M, all of which are bounded. The OEM is in control of N_{ISR} and T_{ISR_N} - both of which can affect IST latencies.

Windows CE also adds restrictions to ISTs: The event handle that links the ISR and IST can only be used in the WaitForSingleObject function. Windows CE prevents the ISR-IST event handle from being used in a WaitForMultipleObjects function, which means that the kernel can guarantee an upper bound on the time to trigger the event and time to release the IST.

Microsoft analysis tools

A number of tools are included with Windows CE that can be used to determine the interrupt timings, application execution behavior, operating system feature timings, and scheduling timing.

- Kernel Tracker: This tool provides a visual representation of the execution of the Windows CE .NET operating system on a target device. This tool can be used to view thread interactions, internal dependencies, and system state information in a real-time environment. For the purposes of this article, the interaction between interrupts, threads, and processes is examined.
- OSBench: This tool enables developers to collect timing samples to measure the performance of the kernel by conducting scheduler performance-timing tests.
- ILTiming. This tool determines the ISR and IST latencies for the target platform. ISR latency is the time from a hardware interrupt to the time of the first Interrupt Service Routine instruction. IST latency is the time from the ISR exiting and the Interrupt Service Thread starting.

Several places within a device can affect real-time performance, including hardware, drivers, and applications. Any thread that requires real-time behavior must prepare all of its resources at startup. That includes the code it runs – it must be non-pageable, and it must also be careful not to share resources (such as critical sections) with non-real-time threads or to call operating system APIs except for those that are defined for use in real-time situations.

Kernel Tracker

The Remote Kernel Tracker can be used to examine the interaction between processes, threads, and interrupts on a running device. Figure 1 depicts an example application that walks the file system of the Windows CE device, one of the folders within a Windows CE operating system image that is mapped to the build release folder on the development desktop machine. The application generates a Kernel Independent Transport Layer (KITL) interrupt for each file residing on the desktop PC. The interaction of the

Intel in embedded.
Roadmap to your future.

>>> Wherever your embedded designs take you, from industrial control and automation to hand-held transaction terminals or innovative medical imaging solutions, Intel provides embedded roadmaps with the ultimate stability you need. Intel offers a broad range of embedded-lifecycle platform building blocks in a choice of extensible product lines based on Intel® Architecture and Intel XScale® technology. Intel's embedded processors and chipsets are backed by a development infrastructure you can rely on to speed time-to-market, supported by a robust hardware and software ecosystem that includes members of the Intel® Communications Alliance. >>> Visit us online, and ask your Intel representative for details. When it comes to embedded stability, now and for the future, your roadmap starts here. www.intel.com/go/industrial

Intel® Communications Alliance

A community of communications and embedded developers and solution providers

intel®

interrupts and application on the running operating system image can be seen, and the time delta between the KITL interrupt and the application thread can also be determined.

In Kernel Tracker, the user interface is divided into three areas, the left pane shows interrupts and processes, the center pane shows the thread/process interaction, and the right pane (not shown) is a key to the symbols used in the center pane. The walktree application is running (at the bottom of the image), but exactly how much time is being spent in the context of the application and the kernel is unknown.

Kernel Tracker provides the ability to set time markers between events and displays the time difference in the status bar of the Kernel Tracker application. There are a number of predefined events, covering synchronization, miscellaneous, and user defined. The thread state is also displayed, which shows threads running, blocked, sleeping, and migrating. In Figure 2, the first time marker is set when the execution of the thread returns from the kernel, and the second time marker is set at the point when the thread context switches back to the kernel.

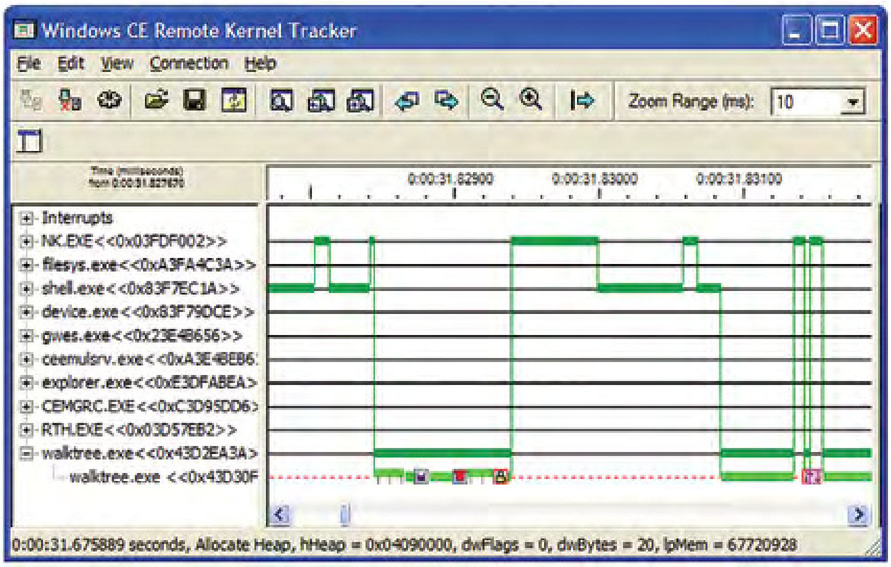


Figure 1

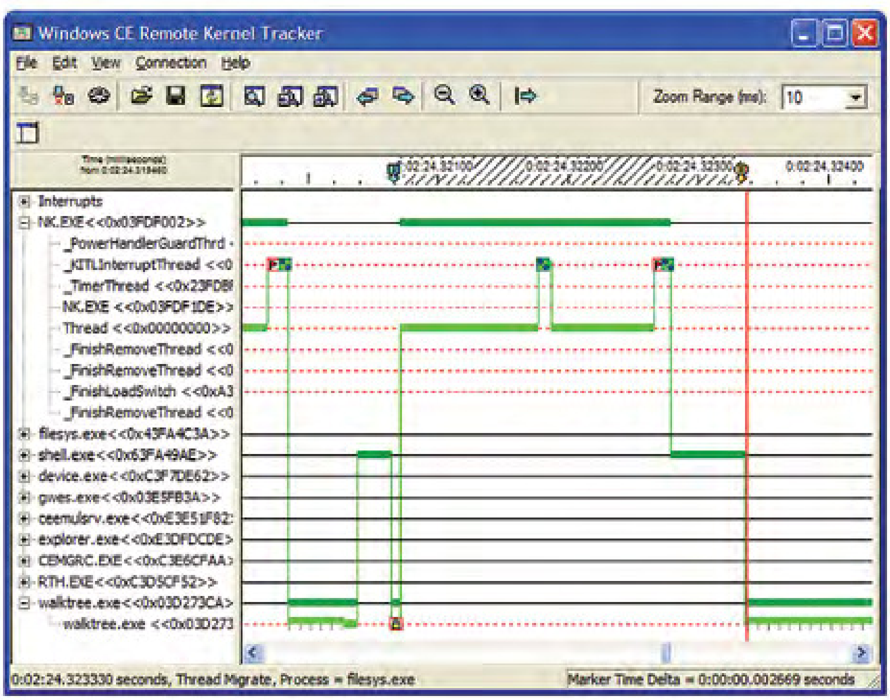


Figure 2

The Kernel Tracker tool can be used to locate and examine deadlock situations, and to examine the amount of time being spent in your application and driver threads. Running Kernel Tracker does affect the overall timing of the operating system, perhaps adding an additional 2 to 3 percent of overhead to the system.

OSBench

OSBench provides benchmarking for an extensive set of operating system conditions from the overhead of an internal Protected Server Library (PSL) call from an application into one of the processes of the operating system (FileSys.exe, Device.exe, and so on). The tests are broken down into seven basic groups as follows:

- CriticalSections
- Event set-wakeup
- Semaphore release-acquire
- Mutex
- Voluntary yield
- PSL API call overhead
- Interlocked APIs (decrement, increment, testexchange, exchange)

As an example, let's look at the output of a couple of tests:

```
==============================================================
| 0.01 | IP = NO | CS = NO | 1 IPS
---------------------------------------------------------------
EnterCriticalSection traditional (blocking) without priority inversion :
Time from a higher priority thread calling EnterCS (blocked) to a lower priority runnable
thread getting run
---------------------------------------------------------------
|   Max Time  =  13.409 µs
|   Min Time  =  7.543 µs
|   Avg Time  =  8.389 µs
==============================================================

==============================================================
| 0.02 | IP = NO | CS = NO | 1000 IPS
---------------------------------------------------------------
EnterCriticalSection fastpath (uncontested)
---------------------------------------------------------------
|   Subtracting out base result of 12 ticks
|   Max Time  =  0.064 µs
|   Min Time  =  0.061 µs
|   Avg Time  =  0.061 µs
==============================================================
```

These tests compare the amount of time it takes to process a call to EnterCriticalSection. There are two paths through that function. The first exercises the critical section with contention and causes the implementation to transition to the kernel to handle the contention. The second remains entirely in the calling process when there is no contention for the critical section and is obviously a lot faster. (This explains why critical sections are preferred for synchronization.)

ILTiming

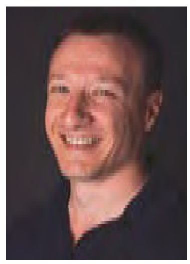
ILTiming measures the Interrupt Latencies in a system. It uses a number of OEM Adaptation Layer (OAL) support capabilities to measure the response times to interrupts for the ISR and IST. These numbers are critical for understanding the constraints of the system, and come as close as possible, with a software-only solution, to measuring the results that can otherwise only be obtained by using an external hardware probe.

Table 1 shows the numbers achieved on an AMD K6 500 MHz-based Windows CE PC (CEPC) system.

	ISR Starts	IST Starts
Mininum	1.6 µs	7.5 µs
Average	1.6 µs	8.4 µs
Maximum	2.5 µs	36.0 µs

Table 1

Windows CE conforms to, and often exceeds, the OMAC definition of a hard real-time operating system, and ships with the tools and resources needed to build, test, and deploy a real-time device. All of these tools – Kernel Tracker, OSBench, and ILTiming – work together to help evaluate the real-time capabilities of Windows CE on a target platform. ECD



__Mike Hall__ is a technical product manager in the Mobile and Embedded Devices Group at Microsoft. Mike has been working at Microsoft for 10 years, originally in Developer Support, focusing on C/C++, MFC, COM, device driver development, Win32, MASM, and Windows CE operating system development, and then as a systems engineer in the Embedded Devices Group. For the last three years, Mike has been in the Embedded Devices Platforms Group, working with Windows CE and Windows XP Embedded. Mike writes a monthly MSDN column on embedded systems development and presents at a number of Microsoft and third party events on Microsoft embedded technologies.

To learn more, contact Mike at:

Microsoft
One Microsoft Way • Redmond, WA 98052
Tel: 425-707-5223
E-mail: mikehall@microsoft.com • Website: www.microsoft.com

Further Reading

Real-Time Behavior of the .NET Compact Framework
Maarten Struys
Michel Verhagen
PTS Software
http://msdn.microsoft.com/library/en-us/dncenet/html/Real-Time_NETCF.asp

Dedicated Systems, Windows CE 5.0 Real-Time x86 Processor
http://download.microsoft.com/download/7/2/f/72fef3b0-9545-46a4-8886-a94f265df9c4/EVA-2.9-TST-CE-x86-01-Iss1.00.pdf

Dedicated Systems, Windows CE 5.0 Real-Time ARM Processor
http://download.microsoft.com/download/7/2/f/72fef3b0-9545-46a4-8886-a94f265df9c4/EVA-2.9-OS-CE-01-I01.pdf

Windows CE 5.0 Real-Time Podcast interview with John Eldridge, Architect, Windows CE
http://blogs.msdn.com/mikehall/archive/2005/09/01/459443.aspx

Real-Time Determinism in Windows CE
http://www.windowsfordevices.com/articles/AT6761039286.html


"Great in a lab, so how about in a UAV?"

Gaining market advantage with an advanced embedded test environment

By Joseph Skazinski

One way to gain market advantage is to build core intellectual knowledge into product design, engineering processes, testing, and customer support. This article describes the market advantages of using an advanced embedded test environment during all phases of product development, product deployment, and product sustaining. An advanced embedded test environment can reduce the testing for a design, analyze performance characteristics, reduce board spins, reduce debugging time and development cost, provide in-field diagnostics when shipped with the product, and reduce the per-unit cost of manufacturing through automated tests running on the processor boards.

Embedded test software can be defined as a software application executed by the processor of an embedded hardware platform that performs various hardware tests. Written and customized to the design parameters of the target processor board, the software application typically varies with each unique board design. The application is loaded into Flash memory, or ROM, of the target platform and typically provides user interaction through a serial port. A faster connection, such as an Ethernet port, can be used once basic board functionality is established.

The test application provides its own boot code and basic Flash memory, serial connection, and RAM tests. Once the basic tests pass, the application switches from a safe boot mode into an interactive user mode. In the interactive mode the user executes a variety of tests using a command line, application program interface or menu system. The test cases vary from low-level system diagnostics to high-level functional and performance tests (see Figure 1).

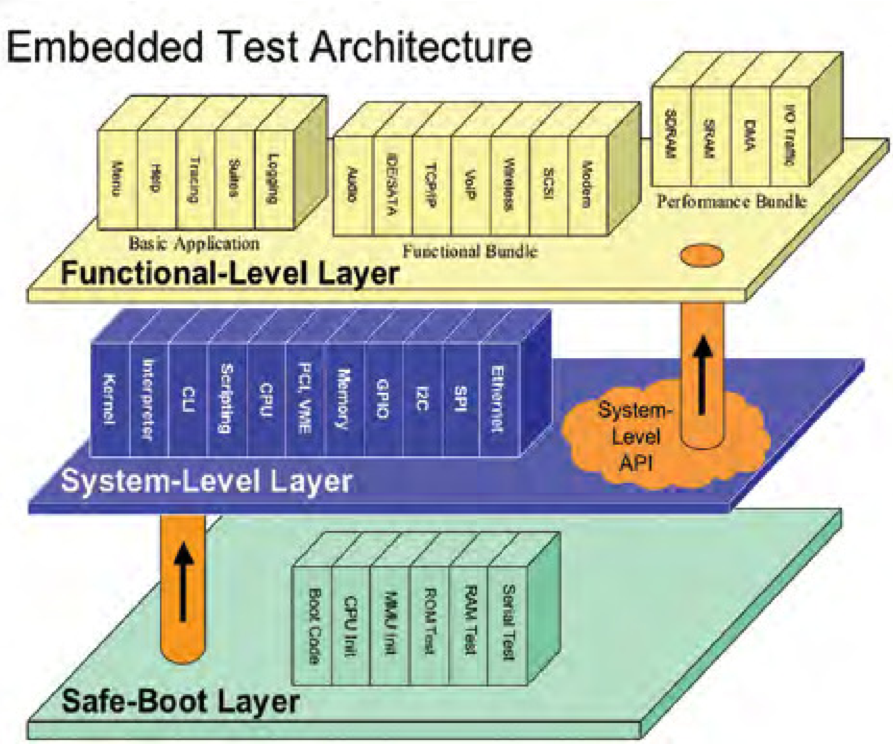


Figure 1

Advanced test environments provide their own kernel, drivers, user interface, canned test suites, execution tracing, test command library, a scripting interface for extensibility, and access to all buses, components, and data interfaces.

Embedded test is deterministic software with easily repeatable test cases that fully exercise all hardware interfaces, buses, and components. To maximize the usefulness of this test software, it should not rely on any third party boot code, loader, or operating system. It must be streamlined and have built-in trace capabilities for localizing and analyzing hardware faults. The test suites should be built using very simple test methods that are coupled into more complex test cases, which are built into an overall component test, and finally included in an easy-to-run system-level test.

Uses for an advanced embedded test environment

Embedded test has proven itself to be the most versatile of all board test environments and complements In-Circuit Testing (ICT) and the use of Automated Test Equipment (ATE) in general. Embedded test proves useful during product development, manufacturing, and product sustaining in the following ways:

Processor evaluation

Most processor companies deliver their latest CPUs using evaluation platform kits. Shipping these kits with an advanced embedded test environment allows designers to get live data with regard to performance and interaction with I/O devices. With an embedded test application designers can explore the system in greater detail without writing new software. The support effort associated with evaluation kits is reduced because a system-level test application easily verifies that the hardware is working properly.

Board bring-up

For prototype testing, often called *board bring-up*, embedded test provides a way to verify that a system is working, as well as to determine whether a design is performing to requirements.

Environmental testing

For environmental testing, an automated embedded test application can easily exercise the various aspects of the system to ensure that the most accurate results can be obtained for characteristics such as power consumption, temperature range, emissions, as well as other regulatory requirements.

Board validation

During application development, embedded test has proven to provide a quick method of assuring that the hardware continues to function as designed, thus aiding in the determination of root cause when system faults are encountered and helping to reduce the phenomenon of finger-pointing between hardware and software.

Manufacturing test

Embedded test provides a clean method of streamlining manufacturing test by reducing human touch-time per unit, providing full access for in-system programming, and providing an efficient test process handoff to an outsourced manufacturer.

Self-test

Embedded test can be reduced in size and functionality to produce a self-test application often referred to as Built-In Self-Test (BIST) or Power-On Self-Test (POST). This simplifies any support effort by providing a known status of the processor board as a starting point. Power-on test results can be used by the application to take appropriate action.

In-field diagnostics

An advanced embedded test application can be shipped with the final product for execution in the field when a system requires the validation that hardware is

operating properly. Access to the diagnostic application is through direct connection to the unit or through a remote interface such as Ethernet. Shipping the embedded test application eliminates the need for support personnel to find the correct test setup or software; it's already in the product.

Board bring-up

After a new board has been designed, it is normal to create a few prototypes. The goal of this activity is to verify that the design will work. If successful, prototype bring-up activity is replaced by environmental testing, delivering boards for application development, sending boards to customers, and verifying that the design will perform to expectation.

Most test methods, such as probes, emulators, and simple embedded debug monitors, will satisfy the first step of verifying that the prototypes work, although minimally. An advanced embedded test environment provides the best solution, and in most cases the only solution, for verifying that the system will meet performance requirements. It also can be used by customers and application developers for hardware testing, and can be used during environmental testing. The advantage of embedded test is that the system is exercised and tested at true processor speeds, and embedded tests run directly on the system, allowing the testing to be shipped to other parties with the system and used in environmental chambers.

Manufacturing test

Using embedded test software for manufacturing test is an ideal solution, but special consideration is required. Manufacturing test differs significantly from board bring-up testing. The main goal for manufacturing test is to provide a simple pass or fail result using an automated process that logs all pertinent data for each board tested. The ideal process is to assemble a board, power it on, and have the manufacturing test application verify the hardware and then serialize it for shipment. The serialization process reconfigures memory with a boot loader, operating system, application image, serial number, and other unique data required by the final application.

Embedded test software runs on the actual hardware platform and has full control of all aspects of the system, such as ability to configure programmable memories with data particular to your product and unique data particular to a unique unit. Embedded test software can be preprogrammed onto

SIDEBAR

Kozio, Inc. has taken the pain out of creating an advanced embedded test environment. Using their patent-pending process, Kozio has case studies available demonstrating the value of their embedded test environment for both board bring-up and manufacturing test; demonstrating that their solutions save significant debugging effort and cost. With more than 250,000 lines of code that have already been validated on dozens of custom platforms, designers get peace of mind that Kozio's solutions will provide test coverage second to none, while freeing up developers for the product's application.

Flash memory so that it is part of the final assembled product producing the simplest test process for contract manufacturers.

As technology advances, the complexity of processor boards increases and the size of processor boards decreases. These changes have led to increased complexity in denser packages with smaller and less accessible access points driving a need for alternate test methods to complement or replace traditional ICT, ATE, and JTAG combinations.

The case against embedded test software

The key arguments against embedded test include the amount of effort and length of time required to write the software. Also, the embedded test software must be written by individuals who have fluent knowledge of both hardware and software. These individuals are potentially a rare and expensive resource. To be truly effective, any functional test software must be developed to the point that allows performance testing, extensive functionality testing, and stress testing. Companies that satisfy these requirements generally hold a distinct market advantage; curiously, embedded test requirements are typically not a stated core competency for most product companies.

Developing the test suite required to test prototypes, exercise the system during environmental testing, provide performance characterization, provide comprehensive testing for manufacturing and deliver in-field diagnostics – all running on the custom hardware – requires a minimum of 18 person-months of development effort for a typical hardware platform. These tasks may pull senior firmware engineers away from other software development tasks required for product release. Writing new software requires that it be tested along with new hardware, further complicating the debugging process for both the hardware and the software.

Summary

Embedded test software provides the greatest test coverage, greatest flexibility, true-speed system diagnostics and performance characterization, stress testing, and reduced touch-time during manufacturing. Embedded test provides the opportunity to ship built-in self-test and in-field diagnostics not available when using processor-based emulation testing. And best of all, embedded test software can be used during all phases of product development and manufacturing as well as product support.

ECD

***Joseph Skazinski** is president and co-founder of Kozio, Inc. Joseph, co-founder Keith Short, and their team launched Kozio in 2002 to deliver the essential embedded software for all embedded hardware platforms using innovative processes. Kozio delivers turnkey advanced embedded system-level test applications for ARM, MIPS, PowerPC, TI, and XScale processor architectures. Joseph holds a BS in Computer Science from Michigan Technology University, and is a member of ACM and IEEE.*

To learn more, contact Joseph at:

Kozio, Inc.
1716 A North Main Street, #127
Longmont, CO 80501-7413
Tel: 303-776-1356 • Fax: 303-776-1357
E-mail: joseph.skazinski@kozio.com
Website: www.kozio.com

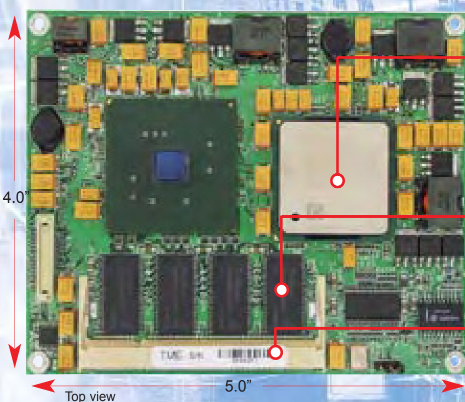
Embedded Computer Solutions For Harsh Environments

-40°C to +85°C Operating Temperature

Five Year Product Availability Guarantee

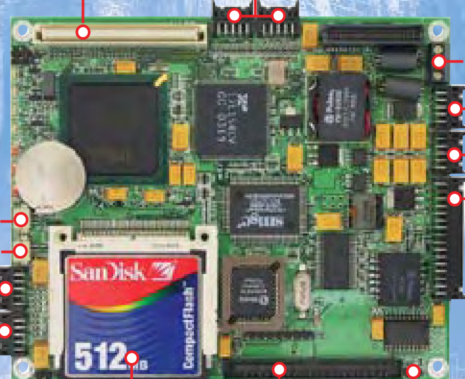
ECM401

World's Fastest Embedded Computer Module



Features:

- (6) Serial ports
- (4) USB 2.0 ports
- (1) 10/100Base-T
- Compact Flash Socket
- E-IDE supports 2 devices
- Video I/F (1600X1200, 32 Mb)
- AC'97 2.2 Audio
- Less than 5 seconds boot up time
- Intelligent thermal management with independent microcontroller
- Power requirement: +12V @ 3A (2.0Ghz P4, 256MB)
- Over 200,000 hours MTBF



Over 15 years in business of designing and manufacturing embedded computer products for OEMs and System Integrators

We can design your Embedded Computer Board in 30 days.

Time-to-market is very critical to a company's success.
TME has a proven development and manufacturing process that allows new embedded computer products, using TME's core technologies, to be developed in 30 days.
OEMs and System Integrators can now bring their products to the market without compromising cost and features at the expense of time.

Typical Applications

- Robotic
- Medical
- Avionics
- e-Kiosks
- Transportation
- Military/Aerospace
- Industrial Automation
- Inventory management
- Point Of Sale Terminal
- Test & Measurement

ECM401 Embedded Computer Module provides all functions and features of a high performance computer on a very small Module.

Embedded Computer Designers can easily design an I/O board with proper connectors and form factor that is most suitable for their applications.

The ECM401 supports up to 1.5Gbytes DDR memory, 2.8GHz CPU, with built-in Audio, Video, USB, Network, serial, LPT interfaces and PCI BUS extension capability.

tme

Toronto MicroElectronics Inc.
5149 Bradco Boulevard, Mississauga, ON., Canada L4W 2A6
Tel: (905) 625 - 3203 ~ **Fax:** (905) 625 - 3717
sales@tme-inc.com ~ www.tme-inc.com

RSC #49 @ www.embedded-computing.com/rsc

Real-time Windows, Firmware

Real-time Windows	
Ardence	**www.ardence.com**
RTX	Real-time extension software that allows Windows XP/XP Embedded to function as a general-purpose operating system and a high-performance, high availability (99.999) operating system, at the same time on the same computer • Guaranteed task scheduling and performance • APIs for direct hardware access • Exception (bluescreen) handling • Windows XP, 2000, and NT API portability • Uniprocessor and multiprocessor support • Application and kernel-level debugging • Applications can start seconds into the boot cycle for extended control.
Argon Technology	**www.argontechnology.com**
Managed PC Boot	A managed PC boot agent (Boot ROM) • Client-based firmware that works with the user's Network Interface Card (NIC) to enable client PCs to access and boot from boot image files located on a network server • Supports multiple boot protocols and network environments, such as the Wired for Management specification's Preboot Execution Environment (PXE), NetWare NCP/IPX (802.2, 802.3, and Ethernet II), IBM's RPL, traditional DHCP, and BOOTP • Allows PCs to perform a Pre-OS boot and accomplish a comprehensive range of management tasks or repairs from a central management station • Suitable for OS and application deployment, critical file backup and restoring, BIOS updates, and virus scanning • After completing tasks, the PC can continue to boot from its local hard disk • Use on diskless workstations enables booting the entire target OS from the server • System Administrators can set up a "Pre-OS boot" in which PCs are booted from a network server before the operating system is loaded from the local hard drive • Can boot off the server in a "network safe mode" if local boot-up fails.
BSQUARE	**www.bsquare.com**
Windows Embedded Studio	A set of development tools used to create custom Windows XP Embedded images based on the requirements of embedded system hardware and software. In addition to providing componentized Windows XP Professional features, the component database included in Windows Embedded Studio also includes embedded-enabling feature components that enable a broad range of device scenarios, such as diskless and headless systems.
TenAsys	**www.tenasys.com**
INtime 3.0 Real-Time Extension for Windows	Combines microsecond deterministic, hard real-time control with standard Windows operating systems • Seamless real-time operation for Windows XP, XP Embedded, 2000, and Windows Server 2003 • Field-proven real-time x86 technology • Supports multiprocessor systems • Real-time TCP/IP and USB OHCI/UHCI/EHCI stacks – both operate independently of the Windows stacks • Full memory protection and address isolation for real-time applications • 256 priority levels and round robin scheduling within each priority level • Mailboxes, semaphores, alarms, regions, and shared memory IPC mechanisms • DeviceNet, PROFIBUS, CANopen, and ControlNet drivers available • GPIB (IEEE-488) and motion control real-time device drivers available.

BIOS/Firmware	
American Megatrends	**www.ami.com**
AMIBIOS8	Scalable BIOS solutions from embedded computing platforms to mission-critical servers • Uses Visual eBIOS (VeB), a Windows-based development environment that reduces the engineering resources required to build a customized BIOS solution • Modular components, called eModules, can be added to an AMIBIOS project or removed without breaking the BIOS • Flexible architecture for embedded and non-traditional computers that do not require a full-featured PC BIOS • Enables desktop, mobile, embedded, and server systems • Platform designers can use server features with mobile hardware • Supports headless system management via a standard serial port using the console redirection and remote Flash eModules.

BIOS/Firmware (continued)	
General Software	**www.gensw.com**
Embedded BIOS 2000	An OEM-configurable firmware layer • Provides the essential functions needed by operating systems and applications for embedded x86 systems • 850+ configurations for building perfect-fit firmware • Build automation for reducing project complexity • Comprehensive set of embedded features • Support for industry-standard initiatives like APM, ACPI, "El Torito" CDROM boot, PCI, plug-and-play, PXE, SMBIOS, SMM, SMP, and Legacy USB • Headless and remote console operation • Burn-in diagnostics • Built-in debugger • Remote access disk and Flash updating • On-target BIOS reflashing • Solid-state disk emulators • Optional multimedia (graphical and/or audio) POST • Standard parts of the architecture provide for OEM-configurable POST, dynamic OEM-written policy at the board design level, chipset, Super I/O, and CPU configuration, and a comprehensive setup screen system • Support for industry standard OS; Linux, Windows family including CE, XP and XPe, QNX, and other Real-Time Operating Systems.
Insyde Technology	**www.insydetech.com**
MobilePRO BIOS	Insyde Technology's system-level software suite includes a core PC/AT BIOS ("boot loader"), and optional value added firmware layers for additional technologies; PCI, plug-and-play, power management, DMI, WfM, USB 1.1 & 2.0, and ACPI support.
InsydeH2O Firmware	InsydeH2O is a fully WHQL-compliant firmware solution that employs a true modular architecture, is written in C code, and uses a driver model that enables a wide range of software engineers to be quickly productive.

Debug and test software	
American Megatrends	**www.ami.com**
AMIDiag Suite	A comprehensive suite of hardware diagnostics used for debugging and testing every facet of PC-based platforms • Detects system components, diagnoses problems, and provides system information about all subsystems • Detailed information about the network environment, sound cards, CD-ROM drives, SCSI devices, power management features, IDE drives, and all other system data can be displayed • Operates in DOS and Windows environments • Compliant with the Common Diagnostic Model standard, and detects and automatically integrates CDM-compliant third-party diagnostic modules that might be already installed • Scripting and advanced multicycle batch capabilities enable the automation of testing for hours, for a complete stress test and burn-in capability.
Kozio	**www.kozio.com**
kDiagnostics	Diagnostic software for embedded designs • Diagnostics include comprehensive functional and performance tests for the processor, memory, buses, peripherals, and other components • XScale processor tests include: Processor exception handling, interrupt tests, instruction and data cache control, MMU setup, core processor register access and coprocessor register access, memory controller tests, peripheral bus tests, timer configuration, DMA controller tests, and CPU reset • PCI/PCI-X tests include: PCI device discovery, including all downstream buses, primitives for generating PCI configuration type 0 and configuration type 1 cycles, performance measurement of data transfers across PCI bus, and PCI compliance tests • RAM tests: Data bus bit tests, address bus bit tests, memory cell tests (verifies that all bits in a memory device can be set and cleared), and random access tests.

Embedded Computing: A VLIW Approach to Architecture, Compilers, and Tools

By Joseph A. Fisher, Paolo Faraboschi, and Cliff Young
Published by Morgan Kaufmann. An Imprint of Elsevier
ISBN: 1-55860-766-8

Reviewed by Jerry Gipper

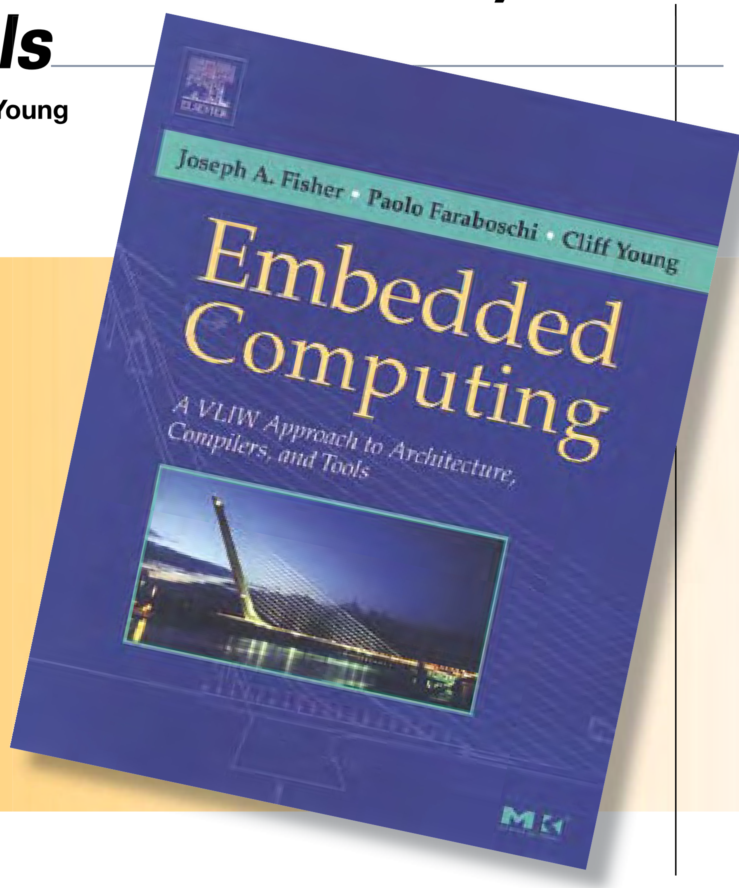


Table of Contents:

- Chapter 1: An Introduction to Embedded Processing
- Chapter 2: An Overview of VLIW and ILP
- Chapter 3: An Overview of ISA Design
- Chapter 4: Architectural Structures in ISA Design
- Chapter 5: Microarchitecture Design
- Chapter 6: System Design and Simulation
- Chapter 7: Embedded Compiling and Toolchains
- Chapter 8: Compiling for VLIWs and ILP
- Chapter 9: The Run-time System
- Chapter 10: Application Design and Customization
- Chapter 11: Application Areas
- Appendix A: The VEX System
- Appendix B: Glossary
- Appendix C: Bibliography

VLIW

With recent advances in semiconductor technology with a feature size of 0.13 microns or less, it is now practical to use Very Long Instruction Word (VLIW) architectures in embedded applications. VLIW architectures are efficient because they replace costly and power-consuming consuming hardware for detecting and scheduling Instruction-Level Parallelism (ILP), with that functionality supplied by a smart compiler.

Fisher, Faraboschi, and Young's book explains – skillfully covering software, hardware, theory, application, and business factors – how such architectures can enable enormous increases in the capabilities of embedded systems. VLIW architectures lend themselves well to the customization and optimization required for embedded applications in ways that superscalar architectures by definition cannot achieve.

Embedded Computing: A VLIW Approach to Architecture, Compilers, and Tools is a great read, engaging and well organized, making it easy to follow and understand. The authors do a good job of telling what is ahead, provide many relevant and contemporary examples, and engage you with generous research and a practical perspective.

Practicing engineers – both chip architects and embedded system designers – will find the techniques they will need to use and develop VLIW-based systems. Instructors will value the rare juxtaposition of advanced technology with practical deployment examples, and students will enjoy the unusually engaging and mind-expanding chapter exercises.

Book flow

The book is set up into roughly three major categories of hardware, software, and applications, but liberally blends the three together. The book clearly expresses the discussion on computer architectures throughout, as none of the three categories clearly stands on their own.

Introduction

Chapters 1 and 2 are introductory in nature. Chapter 1 has a great introduction to the world of embedded processing. Chapter 2 defines ILP, architectural styles including VLIW, and finally discusses how VLIW and embedded domains have merged.

Hardware

Chapters 3 through 6 constitute the *hardware* section of the book. Each chapter expands on the details behind Instruction-Set Architectures (ISA).

Chapter 3 describes ISA, VLIW ISA design philosophy, and particular design issues.

Chapter 4 examines the data structures in modern processors and pays particular attention to how these structures differ in the embedded domain from their general-purpose counterparts.

Chapter 5 explores the microarchitecture or implementation of functionality within a VLIW ISA.

Chapter 6 bridges between hardware and software by discussing System-on-Chip (SoC) methodologies.

Software

Chapter 7 describes current toolchains and embedded- and DSP-specific code transformations.

Chapter 8 touches on a subset of compiler optimizations and transformations in an industrial-strength ILP-oriented compiler.

Chapter 9 covers a broad range of topics from exceptions, application binary interfaces, code compression, operating systems, and multiprocessing in embedded applications.

Applications

Chapter 10 begins by discussing programming languages for embedded applications, and then moves on to performance, benchmarks, and tuning. It also addresses scalability and customizability in embedded architectures.

Chapter 11 visits a number of embedded applications at various levels of detail. Digital printing and imaging and telecommunications are covered at length while automotive, network processing, and disk drives are covered briefly.

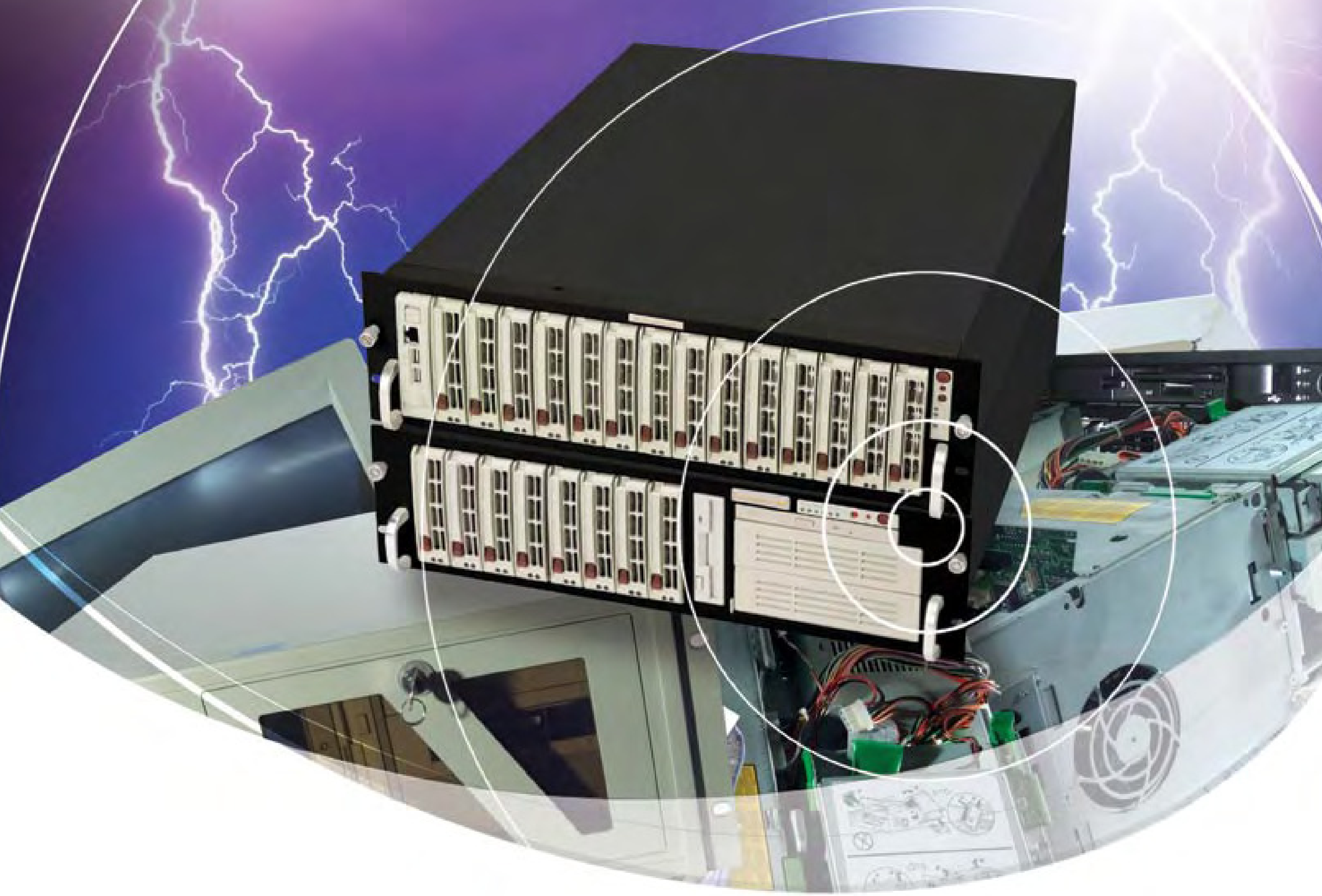
Appendix

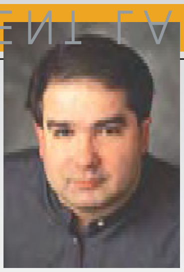
Appendix A is a very thorough example using what the authors call the VEX System. VEX stands for VLIW EXamples. The appendix includes a sample exercise with a VEX instruction set architecture, a VEX C compiler, and a VEX simulation system that can be used as usage examples of the material covered in the book.

Joseph A. Fisher *is a Hewlett-Packard senior fellow at HP Labs, where he has worked since 1990 in ILP and in custom embedded VLIW processors and their compilers. Josh studied at the Courant Institute of NYU (BA, MA, and then PhD in 1979), where he devised the Trace Scheduling compiler algorithm and coined the term Instruction-Level Parallelism. As a professor at Yale University, he created and named VLIW Architectures and invented many of the fundamental technologies of ILP. In 1984, he started Multiflow Computer with two members of his Yale team.*

Paolo Faraboschi *is a principal research scientist at HP Labs. Before joining Hewlett-Packard in 1994, Paolo received MS (Laurea) and PhD (Dottorato di Ricerca) degrees in electrical engineering and computer science from the University of Genova (Italy) in 1989 and 1993, respectively. His research interests skirt the boundary of hardware and software, including VLIW architectures, compilers and embedded systems.*

Cliff Young *works for D. E. Shaw Research and Development, LLC, a member of the D. E. Shaw group of companies, on projects involving special-purpose, high-performance computers for computational biochemistry. Before his current position, he was a member of technical staff at Bell Laboratories in Murray Hill, New Jersey. He received AB, SM, and PhD degrees in computer science from Harvard University in 1989, 1995, and 1998, respectively.*





New OSDL initiative targets current and next-generation converged handsets

By Bill Weinberg

The global mobile phone market is exploding. In Q2004, the market grew 34 percent as almost 700 million handsets made their way from device OEMs into people's hands and onto the global voice and data networks (IDC). By 2009, the global installed base will number more than 2.6 billion mobile phones (Gartner). For the IT industry, these numbers are tantalizing orders of magnitude greater than total shipments and installed base for servers, and far greater in volume than the worldwide desktop market. For the Linux software and related hardware segments, the mobile phone market represents both an opportunity to *break out* and enjoy significant market share in client devices, and to complement the already significant presence of Linux in communications infrastructure.

In 2004 and 2005, Linux has made significant gains as a mobile handset Operating System (OS). Global manufacturers with household brand names like LG, Motorola, NEC, Panasonic, and Samsung today ship two dozen smart phone models based on Linux, complemented by emerging Chinese brands like Datang, e28, Haier, Huawei, and ZTE. Device OEMs, large and small, are choosing Linux as a strategic platform for their smart phones for a mix of technical and economic reasons. On the technical side, they look to Linux for performance, robustness, *gold standard* TCP/IP networking (such as routing), and flexibility. On the economic front, Linux offers OEMs:

- Lower development and deployment costs
- More choice of vendors (including *roll your own*)
- A larger open and commercial technology ecosystem
- An opportunity to unify the divergent and costly product lines and engineering efforts needed to support multiple product tiers (smart phones, feature phones, and entry level devices), network types (GSM, CDMA, analog, and WiFi), and carrier requirements

For all of these strong technical and economic benefits, Linux phones account today for only about 5 percent of the total market. In the fastest-growing smart phone segment (85 percent/year – Gartner), Linux enjoys a stronger position (25 percent in Q2 2005 – Gartner), far ahead of Windows Mobile, PalmOS, or RIM (but behind SymbianOS).

To bolster this very positive trend, the OSDL is creating a new initiative called Mobile Linux Initiative (MLI) to bring together:

- Chipset manufacturers
- Linux distribution and platform suppliers
- Middleware ISVs
- Handset manufacturers
- Integrators
- Carriers
- Operators

On October 17, 2005, in Beijing, the OSDL hosted the launch of this new initiative with a goal of addressing a mix of key technical and economic challenges, *from the kernel up*, to accelerate Linux adoption on mobile phones and other converged voice and data devices. While the ultimate requirements and development efforts will be driven by the members of this new initiative, the following is intended to give readers insight into MLI goals:

Technical challenges

Existing OSDL members and other industry players have repeatedly asked for *Carrier Grade Linux* for the cell phone. What they mean by this request is a specification

for a unified and reliable Linux-based embedded platform that addresses the needs of device OEMs, without creating non-standard *forking* versions of the Linux OS. In particular, they would like to see the following (currently) fragmented technologies become mainstream in Linux with standard implementations in the main Linux kernel tree. What follows is a list of probable technology focuses for MLI, with the ultimate gaps identified and crossed in response to specific OSDL member / MLI participant contributions.

Power management

Today, if portable device manufacturers want to offer a Linux-based and power-managed device, they face a boggling choice among divergent paradigms. OEMs can look to the desktop where notebook-centric schemes like ACPI and legacy apmd dominate, and indeed occupy most discussions of Linux power management on the kernel mailing list. There also exist power/thermal management schemes for Linux-powered blades. For non-x86/IA-32 hardware, OEMs can turn to ARM's own power management framework, or work with the various power management schemes present on silicon from more than 200 ARM licenses (such as Intel XScale or IT OMAP). There also exist unique and further divergent energy conservation protocols from MIPS, its licensees, from Freescale for its CPU lines, from IBM for Power Architecture, from Renesas and Hitachi, and so on across the silicon supplier universe. OEMs can also choose schemes like MontaVista's DPM and other embedded Linux supplier solutions.

While choice is a good thing, too much choice leads to fragmentation. It is thereby very likely that MLI will address the fragmented power management landscape as a Tier 1 requirement as a way to unify the current divergent definitions of this key mobile area.

Radio interface

Companies like Motorola have been building radio sets for nearly a century, and bring this hard won expertise to bear on phone designs, as do other leading players in the mobile marketplace. New entrants, and also new designs from existing suppliers, however, must face an array of daunting design challenges to build a device that meets the requirements of both carriers and regulatory bodies like the FCC, and do so cost effectively. In today's crop of Linux-based smart phones, the GPRS interface resides in an encapsulated *modem* device which can contain an additional CPU core, a DSP, and RF hardware to support wireless communications. Offloading the radio function makes it easier to build a smart phone, but also raises the cost by adding significant components to an already heavy bill of materials. While smart phones offer OEMs sufficient margins to bear this cost, the need for a self-contained modem limits Linux ability to cover the broader market that includes feature phones and entry-level devices.

Some experimental designs today remove the modem and expose the baseband interface to the application OS (as with Nucleus in low-end phones), but doing so exposes Linux to hard real-time requirements that sit at the edge of the response curve of the open source OS (despite huge advances over the last few years in preemptibility). MLI will thereby also likely tackle the best methods for cost reduction through exposed baseband design in terms of real-time response,

context switch, and availability of quality native call stacks on Linux.

Real time

For both the radio interface, and for other capabilities like multimedia, Linux still needs a *nudge* in the direction of RTOS-like responsiveness. Moreover, Linux must meet deadlines and switch context with agility in systems whose clocks can scale erratically to conserve battery power, jumping from 200 MHz peak performance down to 40 MHz (or even 0 MHz) and back in response to system policies and peripheral inputs. MLI will also need to quantify the real-time needs of real-world phone designs, and specify how Linux can best serve those requirements.

aSMP / SoC core interconnect

The current generation of ARM-based phone chip sets feature dense silicon crammed with peripherals to support phone functions such as:

- Display and keyboard controllers
- CODECs
- Power management circuitry
- WWAN and WiFi interfaces
- Flash controllers

These devices can be highly stateful with intricate and hard-to-program shared memory interfaces among them. These channels constitute a troublesome performance bottleneck. MLI will certainly address ways to streamline intercore communications, building bridges among process functions for maximum transparency and performance.

Multimedia

Linux, and indeed other phone Operating Systems, today lack a unified multimedia framework, and exhibit huge challenges for interoperability, API-wise, format-wise, and in terms of a mixture of IP from differing sources. It is in the initial MLI vision to unify the existing multimedia efforts, but to do so in as open and unburdened fashion as possible.

Small footprint

While today's smartphones ship with as much as 128 MB of Flash and 64 MB of RAM, a good phone OS shouldn't necessarily seek to occupy every last byte of available storage. Every bit used by the OS and middleware is a bit not available to OEMs for their value-added content. And, while Linux can deploy in as small a footprint at 1 MB or less, phone configurations loom much larger (such as Microsoft minimum footprints of 28-30 MB). MLI will certainly work with the mainstream kernel community to continue to compress the Linux minimum useful footprint to meet the needs of mobile phones and other embedded devices.

MLI deliverables

When MLI met in October for the first time in Beijing, the new body appointed interim governance and immediately got down to the job of gap analysis. The first likely deliverable will be a preliminary requirements specification, along with use cases and some marketing output. The most important deliverable, as with Carrier Grade Linux, will be the instigation of new open source project work to fill the gaps it identifies, and to bring currently divergent technologies (like those mentioned) into the Linux mainstream.

Stay tuned through 2006 for more developments.

Open Source Development Labs, Inc.
12725 SW Millikan Way • Suite 400
Beaverton, OR 97005
Tel: 503-626-2455 • Fax: 503-626-2436
Website: www.osdl.org

New PRODUCTS

By Chad Lumsden

TABLE OF CONTENTS

ARINC

The MathWorks, Inc.

Website www.mathworks.com
Model: xPC Target Drivers **RSC No:** 23099
Drivers for the company's xPC Target rapid prototyping product • Support hardware from Condor Engineering • Supports both commercial (ARINC 429) and military (MIL-STD-1553) avionics databases • Provides aerospace engineers who use Condor avionics boards with the ability to use Model-Based Design, including xPC Target real-time testing software, for design and testing of avionics systems

AVIONICS

AIM-USA

Website www.aim-online.com
Model: EasyLOAD-615A **RSC No:** 22657
An intuitive software package supporting target data loader operations for use with AIM's AFDX/ARINC664 test and simulation interface modules • All data loader operations performed in accordance with ARINC Report 615A-1/2 and ARINC Report 665-1/2 • Can manually or dynamically FIND/SNIP target end systems and configure the target's ID information, communications parameters, and 615A Data L Load protocol plus TFTP protocol characteristics • Multiple data loader operations with multiple target end systems can be executed simultaneously, including information operation, uploading operation, media defined download operation, and operator defined download operation • For troubleshooting and historical purposes, EasyLOAD-615A can log all client/server communications during data loader operations • Available for Windows 2000/XP

CHIPS & CORES: POWERPC

AMCC

Website www.amcc.com
Model: PowerPC 440GR **RSC No:** 20226
Targeted at networking and storage control plane applications • Based on the PowerPC 440 superscalar core, the low cost AMCC 440GR offers increased processor speed, memory performance and integrated dual Fast Ethernets • Ideal for line card, system control and multi-radio devices, the AMCC 440GR operates at a clock frequency of up to 667 MHz • The AMCC 440GR requires one-third less power than any other member of the 440 family, making it ideal for low power applications • The AMCC 440GR's dual Fast Ethernet controllers function at full or half duplex at 10/100 Mbps • They are configurable as one MII, two RMII, or two SMII ports with a packet reject interface • Along with a 65 percent increase in performance over AMCC's 405GPr, the AMCC 440GR also supports double data rate memory (DDR266) with a peak bandwidth of 1.1 Gbps • Optional ECC support is also provided • The 440GR supports a 32-bit PCI revision 2.2 compliant interface with multiple read pre-fetch and write post buffers and the ability to boot the processor from PCI bus memory • The 440GR's external bus controller provides support for up to 6 ROM, EPROM, SRAM, Flash or slave peripheral I/O devices

RSC #20226

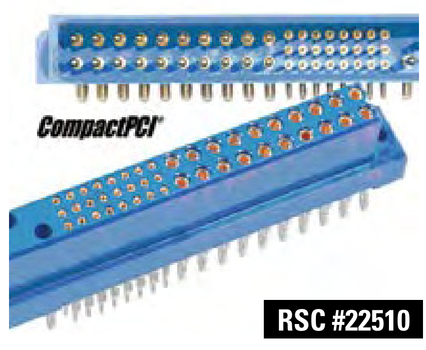
CONNECTOR: BACKPLANE TO POWER SUPPLY

American Conec Corporation

Website www.conec.com
Model: ATCA Series **RSC No:** 22511
Screw machine high reliability contact for the mating side combined with stamped form pressfit zones • Rugged construction • Polarizing system • Premating contacts • Pressfit contacts • Selective loading of contact positions • Screwdown hardware • Special variations on request • RoHS compliant

RSC #22511

Model: CompactPCI Series **RSC No:** 22510
Screw machine high reliability contact for the mating side combined with stamped form press fit zones • Premating contacts in selective positions • Polarizing, coding system • If needed, mounting screws for PCB are available • Selective loading, mixed layout contact configurations • High reliability and longevity • RoHS compliant



RSC #22510

DATA ACQUISITION

Acqiris

Website www.acqiris.com
Model: MAQbox Portable **RSC No:** 22757
A bench-top, standalone solution to multi-channel data acquisition • Eliminates the need for extensive software development • Provides measurement and analysis of many analog signals • Allows user to mix digitizer types within the same system (8-, 10-, and 12-bit modules) • Available in three sizes: MAQbox3000, MAQbox5000, and MAQbox8000 • Embedded processor, 1.6 GHz Pentium, 1 GB RAM, and 60 GB HDD (minimum) • Features AcqirisMAQS, multichannel acquisition software

DEVELOPMENT PLATFORM

VersaLogic Corp.

Website www.versalogic.com
Model: ENCL-5 **RSC No:** 22673
A development enclosure for use with embedded computers in standard PC/104, EBX, and EPIC form factors • Top-mounted location provides full access to SBC and expansion boards • Up to two PC/104 expansion modules can be added • Easy access via front cable connectors • System device containment • Back panel cutout fits standard ATX power supplies, provides access to power switch and cable plug • Light gauge steel construction • Optional lid protects during storage or transport and serves as mounting location for flat panel display up to 14.1 inches

RSC #22673

EMBEDDED PXI CONTROLLERS

Conduant Corporation

Website www.conduant.com
Model: StreamStor Amazon **RSC No:** 22655
400 Mbps of sustained performance • System designers may choose from a wide variety of pluggable interface boards and COTS-based PCI input/output options • Full length 64-bit Universal PCI bus board that controls 16 SATA drives on eight buses • Records in single and multi-channel on the PCI bus • 1 GB onboard memory buffer • 800+ Mbps mezzanine board interface • Onboard PowerPC and real time operating system in standalone or integrated operation • FPDP II pluggable daughter board, backward compatible with FPDP I • Data forking, data partitioning • Simultaneous playback while record/quick erase • Wrap/circular recording • Hardware interlocked data path

ENCLOSURE + CARD RACK + POWER SUPPLY

Hybricon Corp.

Website www.hybricon.com
Model: RME21XC Cooling **RSC No:** 22620
Extreme cooling for up to 100 W per slot • High quality ruggedized construction • CompactPCI, VME64x, VME, VXS backplanes available • Supports full 21-slot backplane with unobstructed rear transition slots • IEEE 1101.10/11 compliant card cage • 1600 W embedded power • Fan speed controller reduces acoustic noise and provides locked rotor failure detection and LED • Supports optional front accessible peripheral devices


RSC #22620

FABRICS: FIBRE CHANNEL

ATTO Technology, Inc.

Website www.attotech.com
Model: 2300E/R/D & 2350C **RSC No:** 22675
Designed for enterprise applications • 1 x 2 with one 2.125 Gigabit Fibre Channel port and two independent Ultra3 SCSI buses • Intelligent bridging architecture for optimized performance • Available in low voltage differential Ultra3 SCSI • Backward compatible with all single-ended SCSI devices • Auto negotiates to 1.0625 Gigabit Fibre Channel • SFP Fibre Channel connectors • Dual-stacked VHDCI SCSI connectors • ATTO ExpressNAV integrated management console eases configuration and upgrades • Serverless backup-supports SNIA extended copy command • Meets Class 2 and Class 3 ANSI Fibre Channel specifications • Full support for FCP2 sequence error recovery • 10/100

Ethernet with RJ-45 connector for LAN local management and monitoring • RS-232 serial port for local management and monitoring • Full support for SNMP, Telnet, FTP, and SCSI Enclosure Services • 2300 E/R/D specific: up to 185 MBps sustained throughput, capable of moving data at up to 660 GB per hour • Available as compact, embeddable board, or desktop enclosure with rack-mount kit • 2350C specific: up to 195 MBps sustained throughput, capable of moving data at up to 700 GB per hour • Available in the industry-standard 4U CompactPCI form factor

FRONT-PANEL HARDWARE

APW Electronic Solutions

Website www.electronicsolutions.com
Model: ATCA Front Panels **RSC No:** 22816
8U x 6HP with optional side 2 covers • Sheet metal construction and extruded aluminum versions available • Dual alignment pins and captive thumb screws • Customized cut-out configurations and finishes available • Standard Ni/Cu emi/rfi shielding gasket with optional BeCu gasket • Fully assembled or supplied in kits • PICMG 3.0 compliant • Standard panel construction is cold rolled steel with a protective zinc plating

Gompf Brackets, Inc.

Website www.bracket.com
Model: Front Panels **RSC No:** 23103
A series of CompactPCI front panels • 3U, 6U, and 9U height and 4HP, 8HP, and 12HP width • Complies with PICMG 2.0 R2.1 and IEEE 1101.10 • Multiple ejector handle options; fiber over foam or silver-plated beryllium copper EMI gasketing • Custom cutout configurations, silkscreening, sticker labels • Hotswap microswitch, 6061-T6 aluminum, clear chromate plating, custom plating and finishing • Aid in design and engineering • 3D models available

INTERNET APPLIANCES

EMAC, Inc.

Website www.emacinc.com
Model: SoM-5282EM **RSC No:** 22997
Coldfire MCF5282 66/80 MHz 32-bit CPU • uClinux with optional real-time support • Telnet, FTP, and HTTP servers • SODIMM processor bus expansion • 10/100Base-T fast Ethernet • 16 GP I/Os & 8 channel, 10-bit A/D • 3 serial ports, CAN 2.0 & SPI • 4.5 MB Flash & up to 16 MB RAM • Typical power consumption < 2 watts • Real-time clock & nonvolatile memory • Programmable in Java or C • Robust, free Eclipse development tools

MASS STORAGE: SATA

Western Digital

Website www.westerndigital.com
Model: WD Caviar RE2 **RSC No:** 22702
SATA hard drive • 400 GB • 7,200 RPM • Rotary Accelerometer Feed Forward (RAFF) technology • Time Limited Error Recovery (TLER) • Native Command Queuing (NCQ) • 16 MB cache • 8.7 milliseconds (ms) average seek time

MEZZANINE: PRPMC

Artesyn Communication Products, Inc.

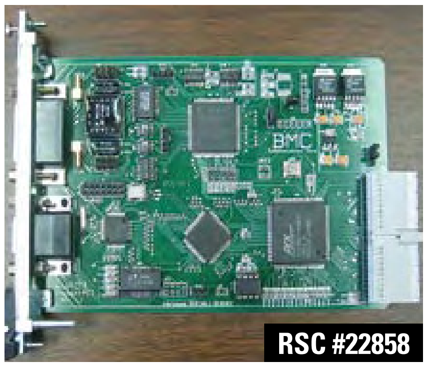
Website www.artesyncp.com
Model: Pm8560 ProcessorPMC **RSC No:** 22783
Suitable way to add protocol processing and signaling capability to telecom systems equipped with PMC or PTMC (PCI Telephony Mezzanine Card) expansion slots • Equipped with 512 MB of SDRAM, 32 MB of Flash memory, and a 10/100/1000 Ethernet port • Optimized for protocol processing applications such as SS7/SIGTRAN signaling and ATM AAL-5 • Operating system support for the Pm8560 includes MontaVista Carrier Grade Linux CGE3.1 • The Pm8560's high performance and versatile protocol support makes it suitable for a wide range of applications requiring multiple E1/T1/J1 interfaces for protocol and packet processing, including VoIP gateways, signaling gateways, base station controllers, radio network controllers, and base transceiver stations • Integrated MPC8560, multichannel T1/E1 interface • Gigabit Ethernet packet interface • PrPMC form factor • Enables telecom OEMs to take full advantage of the MPC8560 and add high-speed packet/protocol processing and signaling to a wide range of telecom systems

For further information enter the product's RSC# at www.embedded-computing.com/rsc

MIL-STD-1553

BMC Communications

Website www.bmccorp.com
Model: cPCI-UADI **RSC No:** 22858
MIL-STD 1553 ARINC 453/708 ARINC 429, 575, 561, 571, 572, ... RS-232/422 protocols on a single board • 1553 modes: BC, RT, BM, BC-RT, RT-BM, and BC-BM • 1553 MRT up to 15 RTs • 1553 multiple store buffers with 32-bit timetag • Two independent ARINC receive transmit channels • Two independent RS-232 receive and transmit channels • ARINC RS-232, baud rate 100 Hz to 1 MHz • Eight I/O and analog inputs with 12-bit resolution • 1553 – Error injection-detection



MOTION CONTROL

Delta Computer Systems, Inc.

Website www.deltamotion.com
Model: RMC100 and RMC70 **RSC No:** 22920
Delta Computer Systems manufactures RMC Motion Controllers for servo-hydraulic and servo-motor applications featuring fieldbus communications, Ethernet/IP, PROFIBUS-DP, Modbus Plus, Serial and Digital I/O (25+ protocols) • Connect I/O with Mix and Match Transducer modules allowing more than 500 configurations. Use powerful RMC software to easily setup, tune, and diagnose your applications

POWER SUPPLY

C&D Technologies

Website www.cd4power.com
Model: UHP-28.2/12-D48 **RSC No:** 22873
DC/DC converter • Up to 12 amps at 28.2 V output • Usable power up to 110 °C baseplate • Half-brick configuration with multiple I/O protection features

PROCESSOR: PENTIUM M

Arbor Technology Co.

Website www.arbor.com.tw
Model: ARBOR ITX-i7435 **RSC No:** 22958
Support Intel Pentium M Dothan Socket 478 up to 2.26 GHz (FSB 533 MHz) • Support the x16 PCI-Express expansion slot • Support two DDRII SDRAM DIMM socket up to 2 GB • Support Dual Display (Independent Display) • Support 18~36 bit LVDS, TFT LCD • Support 2 Serial ATA ports • Support 8 USB2.0 ports

Axiomtek

Website www.axiomtek.com
Model: SBC84810 **RSC No:** 22763
Fanless LV/ULV P-M embedded SBC with DualView and audio • LV Intel Pentium M 1.1/1.4 GHz and ULV Intel Celeron M 1G 600 MHz processors • Low power and high performance • Mini PCI expansion, four COM ports, and dual LVDS option via stacking kits • DualView with different content and resolution • System memory of 1 x 200-pin DDR266 SODIMM max. up to 1 GB • Phoenix-Award 4 Mbit with RPL/PXE LAN boot ROM, SmartView, and customer CMOS backup • CompactFlash Type-II socket with DMA supported • Reset-supported watchdog with 255 levels • 146mm x 104 mm • AC'97 codec audio • 4 x USB 2.0 ports via box-header for wiring solution

Fastwel Co. Ltd.

Website www.fastwel.com
Model: CPC600 **RSC No:** 23008
Full VME64/64x compatibility • Intel Pentium M up to 2.0 GHz onboard, DDR SDRAM up to 2 GB • Four Gigabit Ethernet ports 10/100/1000 Mbps • Two Serial ATA channels via Rear I/O • VITA 31 Specification Conformance • Live Insertion Support • IPMI ready • Operating temperature range from -40 °C to +85 °C • Hardware monitor: voltages and temperature control, system control functions, remote control • 32 MB Solid Sate Disk soldered onboard • Programmable watchdog timer and real-time clock with Li battery • MTBF: 85,000 hours

Newtec Reps

Website www.newtecreps.com
Model: VMIVME-7810 **RSC No:** 22782
Intel Pentium M processor, 478-pin µPGA • One 10/100 Ethernet port on the front panel • IEEE 802.3/802.3u • 100Base-Tx • IEEE 802.3x full-duplex flow control • 3 KB TX/RX FIFO • Two 10/100/1000 Ethernet ports on the front panel • IEEE 802.3 Ethernet interface for 1000Base-T, 100Base-Tx, and 10Base-T applications (802.3ab, 802.3u, 802.3) • 16 KB TX FIFO, 48 KB RX FIFO • IEEE 802.3 Ethernet interface for 1000Base-T, 100Base-Tx • Dual channel SCSI support • One Ultra320 SCSI channel available on front panel • One single-ended channel available through P2 backplane connector • Software RAID support • Two 133/100/66 MHz PCI-X expansion sites

REFERENCE SYSTEM

Pigeon Point Systems

Website www.pigeonpoint.com
Model: BMR-AVR-AMC **RSC No:** 22793
Board Management Reference Design for AdvancedMC modules • Schematics for a complete MMC subsystem, ready for integration into the design of your AMC, with adaptation as necessary • Firmware for that subsystem, delivered in source form and with development tools, ready for simple and quick adaptation to the specific requirements of your product • One-stop support for hardware, firmware, and software used in developing and delivering your IPM Sentry BMR-based MMC • Thoroughly tested with other management components at PICMG AdvancedTCA/AMC Interoperability Workshops • AMC hot swap interfaces (handle and blue LED) • Small footprint and low power: Active components consume only 30 mA of max power IPM Sentry Board Management Reference Design for AdvancedMC Modules (BMR-AVR-AMCm) 2 • Numerous pigeon Point Systems' firmware extension commands, primarily used over the payload and debug serial interfaces • ASCII-based serial interface protocol supported via UARTs to payload processor and serial debug interface

ROUTERS/SWITCHES

Interface Concept

Website www.interfaceconcept.com
Model: ComEth4165 **RSC No:** 22749
A conduction and air-cooled 3U CompactPCI Layer-2 full managed Gigabit Ethernet switch • Built on the latest generation of Gigabit switch engine combined with a PowerPC processor for the switch management • Specifically dedicated to embedded systems in rugged military and aerospace internetworked applications requiring operating temperatures of -40 °C to +85 °C • 9 auto-sensing ports (Ethernet 10/100/1000Base-T) to the backplane (J2 connector) or a Rear Transition Module (RTM) • Auto-crossover, auto-polarity, auto-negotiation, and automatic MAC address management • Wire speed Layer 2 switch-ing, L3 Quality Of Service (QoS) and network management • MAC based authentication, Multicast (static and IGMP snooping) sup-port, VLAN control, STP/RSTP • HTTP server and SNMP interface for an easy manage-ment • Virtual Cable Tester (VCT) automatic cable opens, shorts, impedance mismatch, reporting

PRODUCTS
NEW
PRODUCTS

SERVER

NextCom

Website www.nextcomputing.com
Model: Flextreme **RSC No:** 22828
A briefcase-sized portable server and mobile workstation • Open standards architecture • Processor configurations: single, dual, or dual-core Opteron processors • Multiple high-performance memory configurations • Multiple Gigabit Ethernet ports • Multiple SATA and USB 2.0 ports • PCI Express X16 expansion slot • Optional Ultra160 SCSI or fiber channel • Dual head graphics support • Multiple removable or fixed HDD configurations • Supports several operating systems: Linux, Microsoft Windows 2003 Server, XP Pro, 2000 Pro, Solaris x86, Trusted Solaris x86, Solaris 10, Win-UX concurrent use Linux and Windows, and multiple dual boot environments

SOFTWARE: DEVELOPMENT TOOL

Catalyst Systems Corporation

Website www.openmake.com
Model: Openmake 6.4 **RSC No:** 22613
Comprehensive solution incorporates new activity scheduling and audit features for IBM- and Eclipse-based software development and Perl environments • Improved developer project control through reusable build control files

Jungo Software Technologies Inc.

Website www.jungo.com
Model: WinDriver USB/PCI **RSC No:** 22715
Added a new .NET managed extensions for C++ WinDriver DLL (wdapi_dotnet.dll) • Added a new PLX .NET C# DLL (plx_lib_dotnet.dll) and GUI sample (PLX_Sample) • Added a new USB .NET C# DLL usb_lib_dotnet.dll) • USB: Added sample for the Microchip PIC18F4550 development board • WinDriver USB Device (Windows): Added framework for vendor control requests in the sample and generated code • Solaris: Added API for performing CPU and I/O caches DMA synchronization • Kernel PlugIn (PCI/ISA) on Solaris: Added synchronization APIs, which include spin-locks and interlocked operations, for single and multiple CPU systems

Micro Digital, Inc.

Website www.smxinfo.com
Model: smxFS Portable File **RSC No:** 22962
FAT API provides the standard C library API: fopen(), fread(), fwrite(), fseek(), fclose(), to the application • FAT Path implements the Directory Entry and FAT table structure handler. It supports FAT12/16/32 and VFAT> • FAT Mount implements the mount function for the plugged in device. It supports FAT12/16/32 • FAT Cache implements cache funcionality for data, FAT table and directory entries • FAT Driver Interface uses a unique interface to

integrate device drivers into the file system • FAT Port implements OS, compiler, and processor related definitions, macros, and functions • Available drivers include RAMdisk, USB Mass Storage, and SD/MMC

PRODUCTS NEW PRODUCTS

TCP/IP

Micro Digital, Inc.

Website www.smxinfo.com
Model: smxNET TCP/IP stack **RSC No:** 23023
Works well for small ROM'ed hosts, as well as larger hosts • No disk services are required • It can configure itself after power up using BOOTP • ROM requirements are small and configurable to application requirements • RAM requirements are tunable to the application and vary from about 34KB (PPP) or 60KB (Ethernet) to about 150 KB (including optional packages) for x86 • Use of smxNet with 16-bit processors and small memory is feasible • A no copy operating mode improves UDP and TCP performance • smxNet provides extensive error monitoring and reporting

TEST SYSTEMS

Geotest

Website www.geotestinc.com
Model: GX7300 **RSC No:** 22955
The GX7300 Series mainframes are 20-slot PXI chassis that can accommodate up to 19 instruments as well as a PXI controller (an embedded CPU or a PXI bus expander interface such as the GX799x or MXI-4) • The 3U form-factor provides a compact test system foot print and provides users with the flexibility to employ both PXI and CompactPCI 3U modules • 20 slots – supporting one 3U (embedded) or 3U (remote) PXI controller and 19 3U PXI or CompactPCI instruments • Built-in peripherals (hard disk drive, and a CD-ROM or CD-RW drive) for embedded controller configurations • Innovative forced-air cooling configuration • UUT interfacing provides excellent thermal management in a compact, 4U package • Built-in microcontroller provides per slot termperature monitoring and notification • Dual 800 W power supplies (1100 W optional) • Cable tray, recessed instrumentation, assemby configurations are • UUT interfacing options

VMETRO

Website www.vmetro.com
Model: Link/Protocol Analyzer **RSC No:** 22795
Supports x1 to x8 PCI Express (PCIe) in PCIe card-edge and XMC form factors • Concurrent operation of analyzer, statistics, and protocol checker functions • Ethernet or USB host connection • Display trace data in chronological, logical, and lane views • Data and payload views strip away the protocol for improved readability and productivity

THERMAL MANAGEMENT

APW Electronic Solutions

Website www.electronicsolutions.com
Model: ATCA Thermal Boards **RSC No:** 22808
8U x 6HP x 280 mm or 70 mm depth • Two versions available - one with a side 2 cover and one without • Supplied fully assembled and include dual alignment and safety ground pins, as well as dual captive thumb screws for easy removal from the shelf • Designed to ensure that the airflow across the width of the chassis cannot take the path of least resistance up through any unused slot positions • Sheet metal and FR-4 construction • Standard nickel-copper emi/rfi shielding gasket with optional BeCU gasket available • PICMG 3.0 compliant

VIDEO: DISPLAY

Chassis Plans

Website www.chassis-plans.com
Model: CPR-27201 **RSC No:** 22982
Rackmount LCD keyboard • 1U high • 20.1" LCD, 1600x1200 UXGA resolution • 16.7 million display colors • Picture-in-picture • Multiple keyboard options including Sun and Nema

VIDEO: ENCODER

Aspex Semiconductor Limited

Website www.aspex-semi.com
Model: Accelera HD Encoder **RSC No:** 22959
PC-based plug-in board - PCI-X • Compatible with many third party applications, giving video editing and post-production facilities the chance to inject a performance boost into their current encoding processes • 100% software programmable, customers will be able to use the same hardware to accelerate multiple functions and standards such as H264 and VC1 • Operates at main profile, high level supporting progressive and interlaced input formats

VIDEO: FRAME GRABBER

Aitech Rugged Computer Systems

Website www.rugged.com
Model: M570 **RSC No:** 22713
A camera link frame grabber PMC module • Standard PMC, 64-bit/66 MHz, compatible with the Aitech host platform • Source is Adimec 1000 m stills camera • Camera link interface from the camera to the frame grabber card using 3M connector; 8-bit resolution, 1024 x 1024 frame size, up to 50 frames/sec • PMC connectors support 64-bit PCI bus • I/O PMC connector for external trigger and LVDS connection as an option • Two MB DPRAM frame buffer (up to two frames) • 64 KB for picture processing storage • Picture processing: threshold, convolution, edge (based on FPGA programming) • Compliant with IEEE-P1386.1 standard for PCI mezzanine cards • Three ruggedization levels • VxWorks drivers

WIRELESS: GSM/GPRS

Inova Computers, Inc.

Website www.inova-computers.com
Model: ICP-GSMSER **RSC No:** 22745
3U CompactPCI, GSM/GPRS, GPS communications board • High accuracy: 2.5 m CEP@50%, 5 m SEP@50% • Particularly high sensitivity of -140 dBm • Software compatible to the AT command set according to GSM 07.05 and GSM 07.07 • Additional COM interface for RS-232, RS-485, RS-422 or IBIS standards • Integrated 3V SIM card • Comprehensive software support for the major operating systems • Extended operating temperature option

ADVERTISER INFORMATION

www.embedded-computing.com/rsc

OpenSystems Publishing™

Advertising/Business Office

30233 Jefferson Avenue
St. Clair Shores, MI 48082
Tel: 586-415-6500 ■ Fax: 586-415-4882

Vice President Marketing & Sales
Patrick Hopper
phopper@opensystems-publishing.com

Business Manager
Karen Layman

Communications Group

Patrick Hopper
Vice President Marketing & Sales
phopper@opensystems-publishing.com

Christine Long
Print and Online Marketing Specialist
clong@opensystems-publishing.com

Embedded and Test & Analysis Group

Dennis Doyle
Senior Account Manager
ddoyle@opensystems-publishing.com

Doug Cordier
Account Manager
dcordier@opensystems-publishing.com

Military & Aerospace Group

Tom Varcie
Account Manager
tvarcie@opensystems-publishing.com

Andrea Stabile
Advertising/Marketing Coordinator
astabile@opensystems-publishing.com

International Sales

Stefan Baginski
European Bureau Chief
sbaginski@opensystems-publishing.com

Reprints and PDFs

Call the sales office: 586-415-6500

www.opensystems-publishing.com